Trabajo Fin de Grado

Irn Bru Motor Open Day Demonstrator

Autor/es

Miguel Esteban Fuertes

Director/es

Keliang Zhou (University of Glasgow)

School of Engineering, University of Glasgow
Escuela de Ingeniería y Arquitectura, Universidad de Zaragoza
Junio 2017

Individual Project 4

ENG 4110P

Irn Bru Motor Open Day Demonstrator

Author:
Miguel Esteban Fuertes

Supervisor:
Dr. Keliang Zhou

University of Glasgow
School of Engineering
April 2017

Abstract

This project involves the design and implementation of a power stage that will drive the rotational movement of an Induction Motor that uses a soft drink can as its rotor. The motor is currently being used in an Open Day Demonstrator powered by obsolete and oversized power stage, microcontroller and power supply. The implementation of the new stage and control via a microcontroller entails the enhancement of the efficiency in terms of size of storage, energy saving and control. The Induction Motor demonstrator requires practice facing the public, which leads to the development of a user interface that may well be attractive to the viewers. The user interface will be launched via smartphone, through an app in Android platform and communicated with the stage via Bluetooth connexion.

Acknowledgements

I would like to extend my sincere gratitude towards the people who have helped and supported me during the development of this project. Despite the fact I have taken big efforts in it, it would not have been possible without them.

I would like to thank Mr. Calum Cossar, who gave me the opportunity to do the project that he offered despite the fact of not being able to be my project supervisor. After my proposal for this project, he right away managed to find an alternative supervisor and he let me use his Irn Bru motor and all the necessary equipment.

Secondly, I express my gratitude to my first supervisor Dr. Keliang Zhou for his follow-up of the project during this year and solving all the queries I sent to him. Likewise, I thank the second supervisor Mr. David Moran for giving me some advice about the oral presentation when Dr. Zhou was abroad. The staff working in the electronics laboratory has also helped me with technical issues, in particular Mr. Andrew Phillips, Mr. Alan Yuilland and Ms. Shona Ballantyne.

I would also like to acknowledge the contribution of my flatmates and friends Martin González and Pablo Gómez who have helped me with their knowledge in different areas of engineering.

Finally, I want to express my gratitude to my family who has always supported me and they offered the opportunity of studying the degree I desired, including coming to University of Glasgow for one year abroad; and all the friends that I have made along these years in university.

List of Contents

Abstract I
Acknowledgements II
List of Figures IV
1. Introduction 1
1.1 Objectives and approach 2
2. Induction Motors 3
2.1 Two-phase machines 4
2.2 Frequency converters 5
3. Power Stage 8
3.1 H-Bridge 8
3.2 Other components 11
3.3 Printed Circuit Board (PCB) 12
3.4 Circuit Assembly 13
4. Sinusoidal PWM wave 15
4.1 Unipolar SPWM 15
4.2 Digital Implementation 17
5. Software implementation 20
5.1 Mbed program 20
5.2 Bluetooth serial communication 23
6. Circuit behaviour 25
6.1 Resistor load 25
6.2 Motor load 25
7. User interface 27
7.1 Android user interface 27
8. Conclusions and further work 30
References 32
APPENDIX A. Former devices 36
APPENDIX B. Schematic and PCB files. 38
APPENDIX C. Mbed Microcontroller Code. 42
APPENDIX D. User interface Java Code 45

List of Figures

Figure 1.Two-phases electric system. [10] 4

Figure 2.Irn Bru Motor 5

Figure 3. Frequency converter circuit. (a) Diode Bridge Rectifier + Filter. (b) Inverter. [13] 6

Figure 4. (a) L6203 Block Diagram. (b)Pin connections. [19] 10

Figure 5. (a) Mbed NXP LPC1768. (b) Bluetooth Module HC-05. 12

Figure 6.PCB.... 13

Figure 7.Circuit Assembly 14

Figure 8.Unipolar Sinusoidal PWM [27] 16

Figure 9.Simulation of SPWM in Matlab. (a) 10 steps. (b) 50 steps. (c) 50 steps two-phases 19

Figure 10.Flow Chart 22

Figure 11.Asynchronous 8-N-1 transmission. 24

Figure 12. Oscilloscope screenshot for resistor load. (a) 50Hz current. (b) 20Hz current. 25

Figure 13.App Layout 28

Figure 14. Final devices. 30

1. Introduction

The Irn Bru Motor is an induction motor and to drive and control it, there are many techniques. The most efficient method to control the speed of rotation is through frequency inverters, a design that converts either alternating current or direct current into another alternating current that we can control its frequency, what means the speed of the motor can be controlled. This is the method chosen for the development of the Irn Bru power stage.

Frequency Inverter circuits transform direct current signal into an alternating signal with a specific frequency. In recent years, many applications related to this operating principle are emerging, what means interested companies are focusing their research on this technique.

Driving electric induction motors through frequency inverters (or frequency converters) is a relatively new solution widely implemented in industry. For this reason, there is still much to research and investigate on these applications to make them become more attractive and efficient in terms of energy and costs. [1]

Inverters play an important role in terms of clean energies and ecological transport. Electrical and hybrid vehicles, which use this technique to drive their motors, are a challenge that is focusing a large sector of researching in transport industry. "For systems with a higher degree of electrification, a proper design of the power electronic subsystems gains importance due to restrictions on efficiency and available volume." [2] Regarding energy production, inverter circuits are necessary to adapt the generated power into the consumption main supply. Solar energy is now increasing with an annual growth rate of 25% and it has exceptional benefits apart from being eco-friendly, such as low installation costs. "The micro-inverter has attracted recent market success due to unique features such as lower installation cost, improved energy harvesting, improved system efficiency, and Plug-N-play operation".[3]

Furthermore, transforming a DC wave into an AC with a desired frequency is also an interesting technique in other domains apart from industry sector, such as domestic usage. Inverter circuits are the operational principle of High Efficiency Illuminating Systems that use LED lighting at different frequencies. "In LED applications, variation implies lamp flicker. Flicker at 100 Hz or 120 Hz is one of the inherent disadvantages of conventional magnetic-ballast fluorescent lighting, and is to be avoided in new technologies." [4] Moreover, in recent years high frequency induction heating, the system used in induction cook tops, is becoming popular due to its efficiency, safety and reliability. [5]

1.1 Objectives and approach

Glasgow University has an Open Day Demonstrator of a motor using an Irn Bru can as its rotor, but its electronic units and controller to rotate the motor are oversized. The aim of this project is to design and construct a smaller power stage that can fit inside the motor box, and develop a user interface via pc or tablet in order to make the unit more portable and easier to set up.

The Irn Bru rotor motor is a two-phase induction motor that uses AC power, so an inverter per phase is needed to generate the AC power from a DC source and rotate the motor. The inverters used in the demonstration are oversized and outdated, and so is the microcontroller. In APPENDIX A are included some pictures of the former devices with their respective dimensions. Nowadays there are updated technologies that let us construct a model with much smaller size. They use microcontrollers ten times smaller than the one being used, or inverters designed with small integrated circuits. In addition, these new technologies are cheaper and more efficient than the old devices.

The output voltage range of electronic motor speed controllers are often produced through inverter circuits (Section 2.2). The DC power for the inverter can be generated from a DC source of any voltage or from an AC wall outlet using rectifier circuit. A 30V DC adaptor from the main supply will be used to make the motor rotate. The inverter circuit has four transistors per phase (H-bridges) that switch in accordance with some pulses, generated by the microcontroller (Section 3.1). The microcontroller will give pulses to each switch in a specific pattern to obtain the output desired. The way of switching that is going to be used in this project is a PWM wave, generated from the microcontroller selected. (Chapter 4)

The electronic circuit constructed and tested will later be redesigned by computer in order to be replaced by a PCB (Printed Circuit Board) (Section 3.3), which will allocate the electronics units more efficiently and reducing the size of the ensemble, a crucial task considering that the purpose of the project is to find a more portable solution for the device.

This project will also require a user interface (Chapter 7) because besides making it easier to control the motor speed, the public might find it more appealing while it is being used in the Open Day demonstration. Without losing the essence of an induction machine, by creating an attractive user interface in a tablet, designing a gorgeous app might make the audience feel more interested about the subject.

2. Induction Motors

Induction motors are electrical machines that can make a rotational movement creating a magnetic field in the stator winding through an AC power supply. Induction motors can be classified by their rotor electrical connections. They can be made without connections in their rotor, as the Irn Bru motor is, or with connections, such as the squirrel-cage type or the wound type. [6]

Induction motors represent the majority of the electric motors used in industry, and about one third of the energy consumed in the world is intended for driving induction motors such as fans, compressors, elevators, pumps or different machinery. [7] Moreover, the majority of induction motors in industry are three-phase induction motors because they provide better characteristics in terms of efficiency, self-start and control.

They are named asynchronous machines (or asynchronous motors) because they rotate at a frequency lower than their synchronous frequency. Synchronous frequency is the frequency of rotation of the magnet field created in the air gap produced by the alternating current in the stator windings. The slip and the number of poles of the motor play an important role in the speed of the motor. The synchronous speed (Ns) is given by the frequency of the magnetic field divided by the number of poles per phase.

$$N_s = \frac{120*f}{p} \text{ [RPM]} \qquad (2.1)$$

The slip is the relative motion between the magnetic field and the rotor. It is expressed as a percentage and it is given by equation (2.2);

$$S = \frac{N_s - N_r}{N_s} \qquad (2.2)$$

where Nr is the rotor speed and Ns is the synchronous speed. Thus, the real speed (in RPM) of the rotor is given by:

$$n = \frac{120*f}{p}(1-s)\text{[RPM]} \qquad (2.3)$$

Analysing equation (2.3), we can actuate in three parameters to vary the rotational speed of the motor: slip, frequency and number of poles. It is difficult to adjust the speed by actuating in the slip of the rotor, and it will involve larger energetic losses. Changing the number of poles was a technique developed for early inductions motors, which were designed with different combinations of poles controlled by switchers. This technique is now obsolete and only offers a selected range of fixed rotational speeds. [7]

Regarding the frequency of supply, frequency converters are the most common and efficient method of controlling the speed. This technique consists in changing the frequency of the main supply to a desired current to supply the terminals of the motor. This one is the technique that is going to be implemented in the development of the project (Section 2.2).

2.1 Two-phase machines

This technology was used in power distribution system of the early 20th century, but it was fast replaced for three-phase electrical power machines that are more efficient.

Two-phase induction motors have an asset above single-phase ones: they can be self-started. On the other hand, they have a disadvantage in comparison with three-phase motors, which can conduct overall amount of power using less conductor mass. [8] The point in that is that they use four wires (two per phase) and those could better be replaced for three wires, as it occurs in three-phase electric power. [9]

In two-phase electric systems, the phases have a 90^{o} phase delay (Figure 1). By having a delay of 90^{o} between phases, two-phase motors can create the torque necessary to start rotating by themselves. To rotate them in the opposite direction, it is only needed to swap both phases. In this project that is going to be implemented in the microcontroller by bringing forward 180^{o} just one of the two phases, so that the final wave form is the same as swapping both of them.

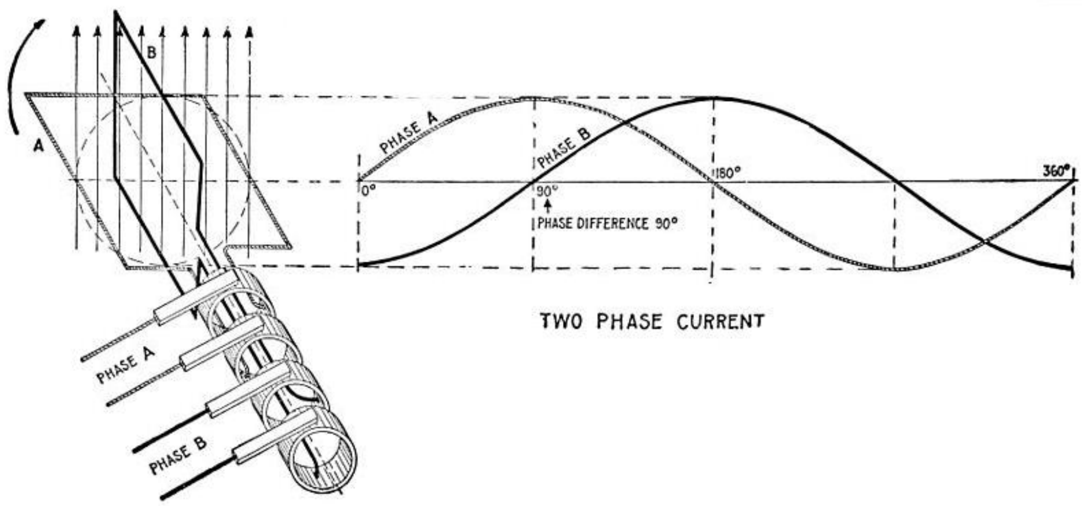


Figure 1.Two-phases electric system. [10]

Frequency inverter circuits to drive two-phase motors are not conventional, they are not used in industrial applications were three-phase motors (and single-phase motors in selected tasks) ac-drive technology is mature today. The reason is that the large majority of two-phase machines are not driven in speed variation; they rotate with a fixed frequency. Industrial pumps, compressors or fans, they all operate at a fixed speed. [11]

The Irn Bru two-phase Induction Motor is shown in Figure 2.

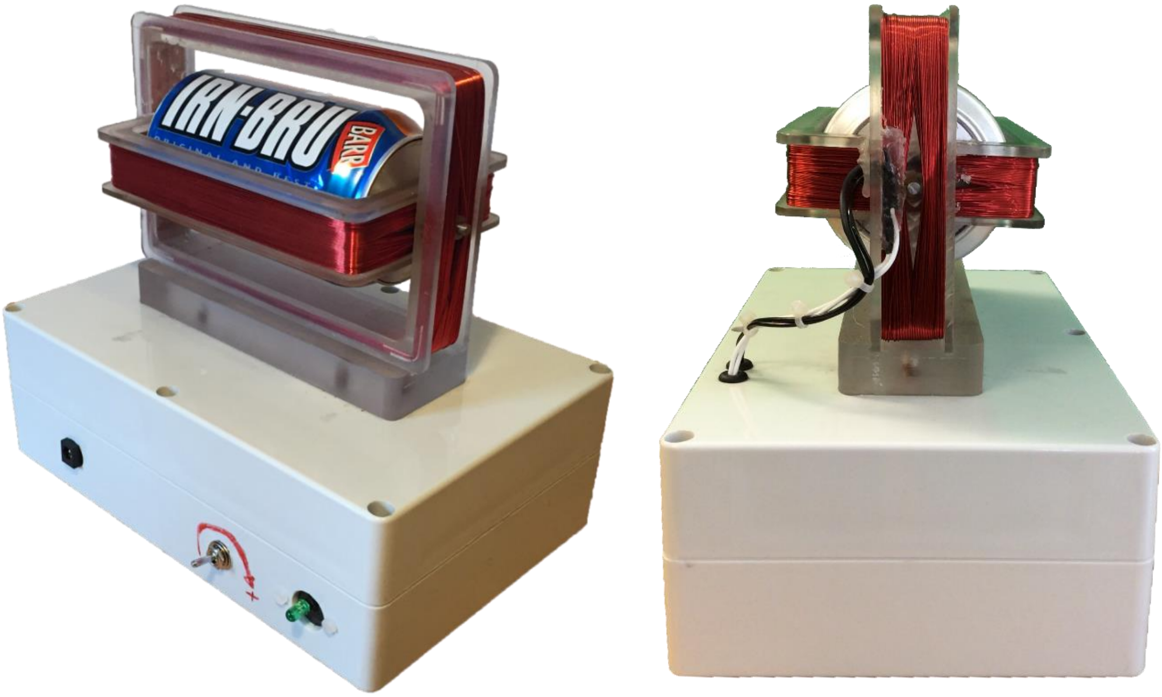
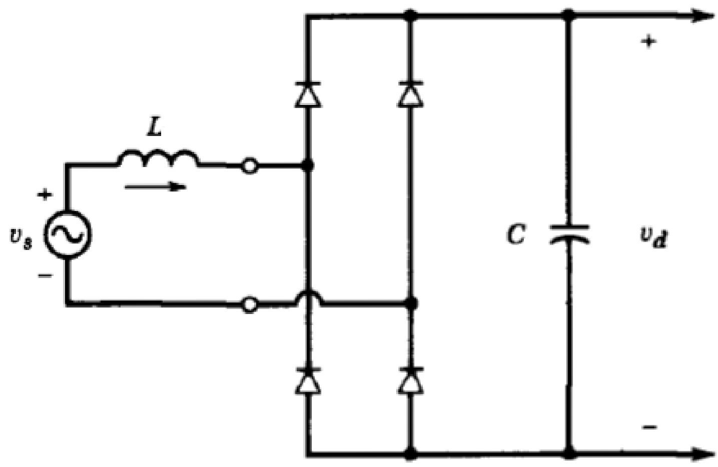


Figure 2.Irn Bru Motor

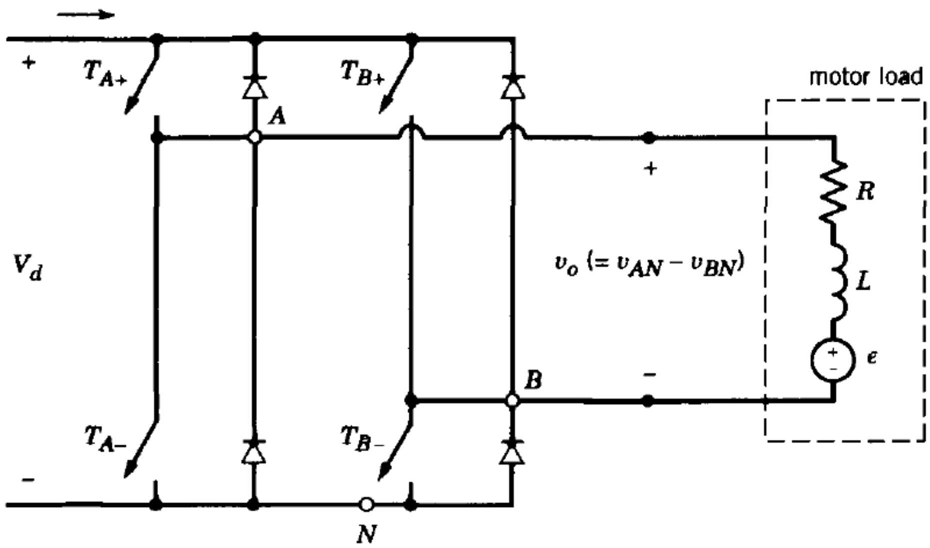
2.2 Frequency converters

Frequency converters, as mentioned before, can transform an alternating wave at a fixed frequency into an alternating wave at a desired frequency. They are mainly consisted of two parts: Rectifier circuit (plus filter) and Inverter circuit.

The Rectifier circuit transforms the AC from the electrical grid into a DC. The trend is to use diodes, which block negative current, to construct the rectifier uncontrolled circuits because they are inexpensive components that provide a good performance. The direct current is enhanced with the coupling of a capacitor that will smooth the continuous current.[12]



(a)



(b)

Figure 3. Frequency converter circuit. (a) Diode Bridge Rectifier + Filter. (b) Inverter. [13]

The Inverter circuit is supplied by the DC output from the rectifier, and transform the DC into AC. It is composed by an H-Bridge of four transistors per phase, in our case eight transistors. Bipolar junction transistors (BJT) or Silicon controlled rectifiers (SCR) were the transistors typically used in first electronic driven frequency inverters. BJT and SCR can drive larger loads, but they have constrains regarding their switching frequency. That is the reason for modern inverters in industry to use Insulated Gate Bipolar Transistors (IGBT), which can switch at much higher frequency, a key characteristic for generating sinusoidal waves. However, they do not offer the same advantages over BJT in terms of driving larger loads, but the fact of being voltage-controlled instead of current-controlled offers more flexibility to be controlled by logic signals. Those transistors are grouped in two pairs per phase, and those pairs switch together to generate the alternating current. The frequency of the resultant sinusoidal wave defines their speed of switching. The H-Bridge theory and the resultant sinusoidal wave generation will be further addressed with more detail in Section 3.1. [14], [15]

The switching of the transistors is controlled by electronic logic. The quick progress of electronics has let induction motors being driven at different ranges of velocity through frequency converters instead of driving them at the main supply fixed frequency. When the frequency is varied just using digital electronics (such as a microcontroller), they are called static frequency converters. This is the reason of frequency converters gaining importance in industry, regarding the fewer expenses using simple electronic devices and the flexibility, reliance and capacity of control they can provide.[1]

3. Power Stage

The power stage design and assembly involves a large section in the global contents of this project. It needs to achieve the objectives mentioned in the introduction, such as the size reduction to fix it inside the motor box. It must also fulfil the requirements demanded by the motor. Initially, the specifications of the motor work characteristics given were experimental values of voltage and current consumption in previously practices. The motor driven with the former devices used in the Open Day Demonstrator was powered with a variable DC source that usually fed the motor at 40 V (Figure A. 1). The current consumed by the motor was not given as a specification. As it will be addressed in Section 6.2, the final amplitude of the output sine wave will be 30 V because it is voltage enough to drive the motor, consequently the input DC voltage necessary to supply into the power stage will be 30 volts.

The power stage can be divided in two sections. One is the driving load section, mainly composed by two H-Bridges, one per phase, and tasked to generate the output signal. The power source voltage supplies this part. The second part is the control part that will send the logic signals to drive the H-Bridges, and it is supplied by 5 volts. Its core component is the microcontroller and it also includes other communication devices such as the Bluetooth module.

3.1 H-Bridge

The H-Bridge, or full bridge, is the essential component of the whole power stage. The structure of a single-phase full-bridge is four transistors, grouped in pairs, and four protector diodes connected in antiparallel with the switches, as it was shown in Figure 3.b.

Regarding Figure 3.b, the H-Bridge consists of two terminals that will supply the ones in the motor, A and B legs. Each leg has a couple of transistors and their corresponding diodes. The switching transistors work in a way that when one of both is turned on, for instance transistor A+ , the other is turned off, A-. The two transistors of each leg are never turned on at the same time. Furthermore, A+ and B- transistors usually work as group, always in the same state, and so do B+ and A-. The switch on of A+ and B- generate the positive component of the sine wave and the other two generate the negative one. Transistors in the same leg (A+ and A- or B+ and B-) must never be switched on at the same time because that means generating a short-circuit in the power supply. [16]

The diodes function is to protect the transistors from a delayed current of the motor when all of them are switched off. The motor current has nowhere to flow if the transistors are off, thus the voltage will rise until that current has a path to follow. That means it will rise until it reaches voltage enough to come through the transistors, causing the death of them. Therefore, by allocating antiparallel diodes, we provide a path for that current to be discharged. [17]

The selection of the proper transistor is a difficult task that will need to take into account many requirements. The bipolar transistors BJT where the only power transistors used until MOSFETs and IGBTs came along. The bipolar transistor is turned on with a large base current while the MOSFET is not controlled by the current through his gate but his voltage, which is an advantage when controlling it from a microcontroller. The MOSFET can switch at higher frequencies than the BJT but last one offers better performance when driving higher loads. The IGBT offers cross properties between BJT and MOSFET transistors, combining conduction characteristics of BJT and voltage control like MOSFET. As mentioned in Section 2.2, the majority of inverter circuits that drive induction motors use IGBT transistors because they offer high speed switching (not as high as MOSFET but high enough to drive a motor) and medium load driving. Newest IGBT transistors offer similar characteristics to MOSFETs in switching so there is always a discussion in which transistors to use. MOSFETs are a mature product in industry but IGBT are still growing and improving their characteristics. IGBT are used in applications that require higher temperature and maximum current working, while select applications at higher frequency will tend to favour the MOSFET transistors. However, MOSFETs under typical conditions can be found usually at lower cost, what make them attractive to manufactures and that is why we can find several H-Bridge integrated circuits that work with MOSFET transistors. [14], [18]

The H-Bridge, as previously addressed, can be built with four transistors and four antiparallel diodes; but as they are a popular product in the electronics market, some manufacturers offer their own H-Bridges implemented into small integrated circuits, designed to drive motor loads. The power stage to be designed requires a reduction of space, so apparently these H-Bridges integrated circuits can be a good choice. The main disadvantage of these devices is that they have a specific design to drive a motor, which entails transistors cannot be controlled individually but in pairs, so they do not offer the flexibility of controlling that four individual switches offer. As we are using the H-Bridge to drive a motor, it will not be a disadvantage but an advantage. Another disadvantage can be heat dissipation, because four different heat sinks can be allocated with individual transistors and only one can be on the IC.

Regarding the requirements of our motor the L6203 DMOS FULL BRIDGE DRIVER is the H-Bridge more suitable for the power stage. It has four MOSFET transistors in place and antiparallel diodes. The transistors can operate at supply voltages up to 48V and it can admit a maximum total RMS current of 4 A. In the block diagram of Figure 4.a is shown the internal structure of the device with a total of 11 pins. [19]

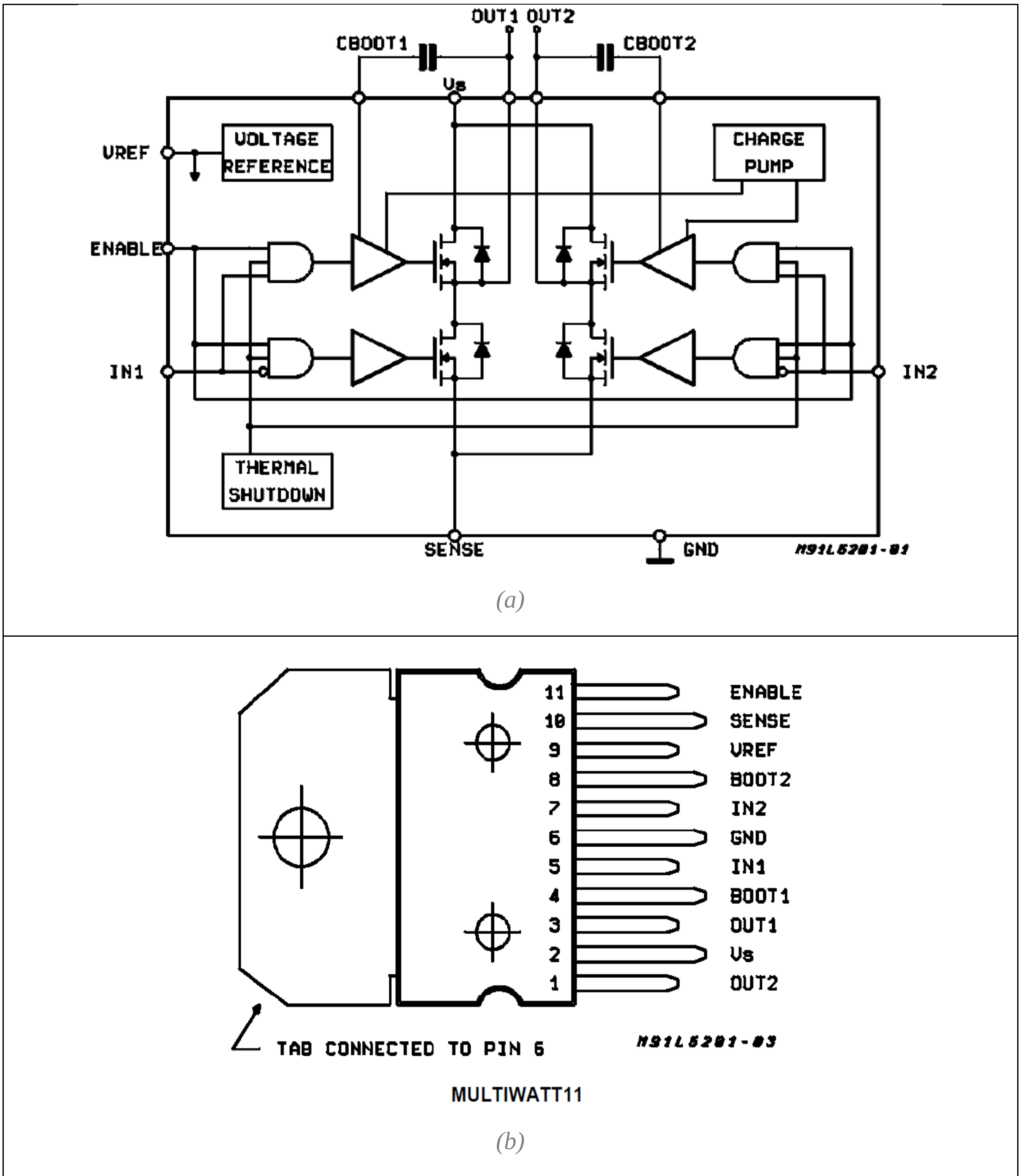


Figure 4. (a) L6203 Block Diagram. (b)Pin connections. [19]

The terminals of each phase of the motor will be connected to OUT1 and OUT2 pins. These pins pull out the alternating signal with amplitude equal to the DC powered through the pin Vs.

The IC has three different inputs: ENABLE, IN1 and IN2. The ENABLE pin will be used to input the PWM signal to the transistors and the IN1 and IN2 will allow activating them. If IN1 is set high then the pair of transistors A+ and B- of Figure 3.b will drive the PWM signal; and when IN2 is high A- and B+ will do it. It is important that IN1 and IN2 pins are never activated at the same time when the PWM signal is being introduced through ENABLE because it would generate a short-circuit. The input pins are set high when they receive a voltage between 2 and 7 volts. [19]

3.2 Other components

It was mention before the power stage is divided in the subset tasked to drive the motor, in other words, generate an alternating 30 V DC wave; and the subset commissioned to drive the switchers from the first one.

Apart from the two H-Bridge implemented in L6203 mentioned in the previous Section3.1, other components shape this subset. Following the advice of the fabricant given in the datasheet of these integrated circuits, one bootstrap 15nF capacitor must be connected between OUT1 and BOOT1 pins to ensure efficient driving of the upper POWER DMOS transistor, and consequently, one between OUT2 and BOOT2 of each H-Bridge. In addition, a 220nF capacitor is recommended to be connected between grounds and pin VREF. [19]

The subset commissioned to drive the switchers sending them PWM signal from a microcontroller. The microcontroller selected is Mbed NXP LPC1768 (Figure 5.a). It is suitable for generating the signals required and was recommended by the laboratory technicians because is the one they currently use and are familiar with. It can be supplied between 4.5 V and 9 V, and 5 V is the typical powering voltage used. It uses C language, its own Mbed libraries, and it is programmed via online in its own compiler.[20] The power stage is going to use a unique power source of 30 V DC (the selection of it is addressed in Chapter 5), meaning that voltage regulator must be included to supply the microcontroller. The voltage regulator selected is ST L7805CV and it supplies 5 V and maximum current of 1.5 A from voltage up to 35 V. Following the datasheet specification, a capacitor of 330nF is coupled between the input and ground; and a capacitor of 100nF is coupled between the output and ground. [21],[22]

All the capacitors used in the circuit are ceramic capacitors because they are cheaper than electrolytic one and they offer a similar performance in medium frequencies.[23]

The communication with the external devices is established through a Bluetooth Module HC-05 that will also be supplied at 5 V (Figure 5.b).

(a)

(b)

Figure 5. (a) Mbed NXP LPC1768. (b) Bluetooth Module HC-05.

Some security components have also been included. A manual switch connected before the 30 V bus blocks or allows the stage to be powered; and a green LED will light warns the user when the switch is on. The LED has a forward voltage of 2 V and a rated forward current of 20 mA, so the resistor selected to connect between the 5 V bus and the LED is 150Ω. Other security components are the two fuses, 1A protection for the whole power stage and 500mA protection for the microcontroller and Bluetooth module. Finally, 100kΩ resistor is connected between each output from the microcontroller and the ground to offer a path for discharge a possible current coming from the MOSFET transistors, whose gate can act as a small capacitor and accumulate small loads. [22] The connections and design of the whole circuit is represented in an schematic created in *Cadence Capture CIS Orcad* in Figure B. 1, APPENDIX B.

3.3 Printed Circuit Board (PCB)

To reduce the space of the power stage and make it fix inside the motor box, we need to optimise the dimensions of the circuit. Printed circuit board, PCB, is a thin board that electrically connects the components of our circuits via pads of copper, allowing signals to be routed between physical devices avoiding the usage of wires. The components are soldered into the board to ensure the electrical connection with the pads and the mechanical attach to it.

Apart from size optimisation, better electrical connection and mechanical attachment to the board, the implementation of a PCB has other advantages over prototyping circuits. The components are organised and distributed in sections for a better understanding of the circuit, and they are placed in a way that electrical noise from other components or pads is minimised.

The PCB has been designed using *Cadence PCB Editor* Software, transferring the schematic file of the circuit previously created in *Cadence Capture CIS Orcad*.

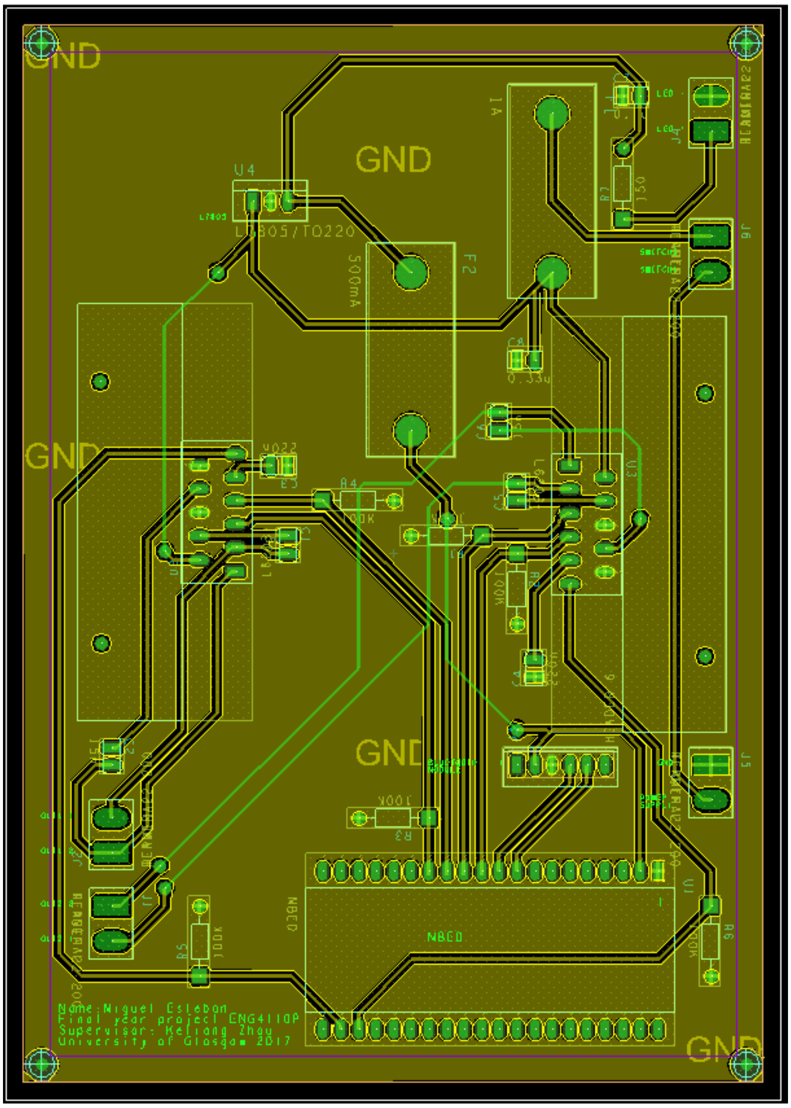


Figure 6.PCB

In Figure 6 it is represented the schematic of the PCB final design. The upper layer is where the components are placed. The lower layer surface is assigned as the ground connection. The production files of the PCB can be found in APPENDIX B, Figure B. 2 and Figure B. 3.

3.4 Circuit Assembly

The components previously defined are soldered in the printed circuit board and this one is fixed inside the motor box. The final circuit (Figure 7) dimensions are 110 x 160 x 50 mm, successfully achieving our objective of size reduction. In the final circuit some heat sinks have been allocated on the integrated circuits to dissipate the heat generated and guarantee a good behaviour of the circuit, as it is addressed in Section 6.2.

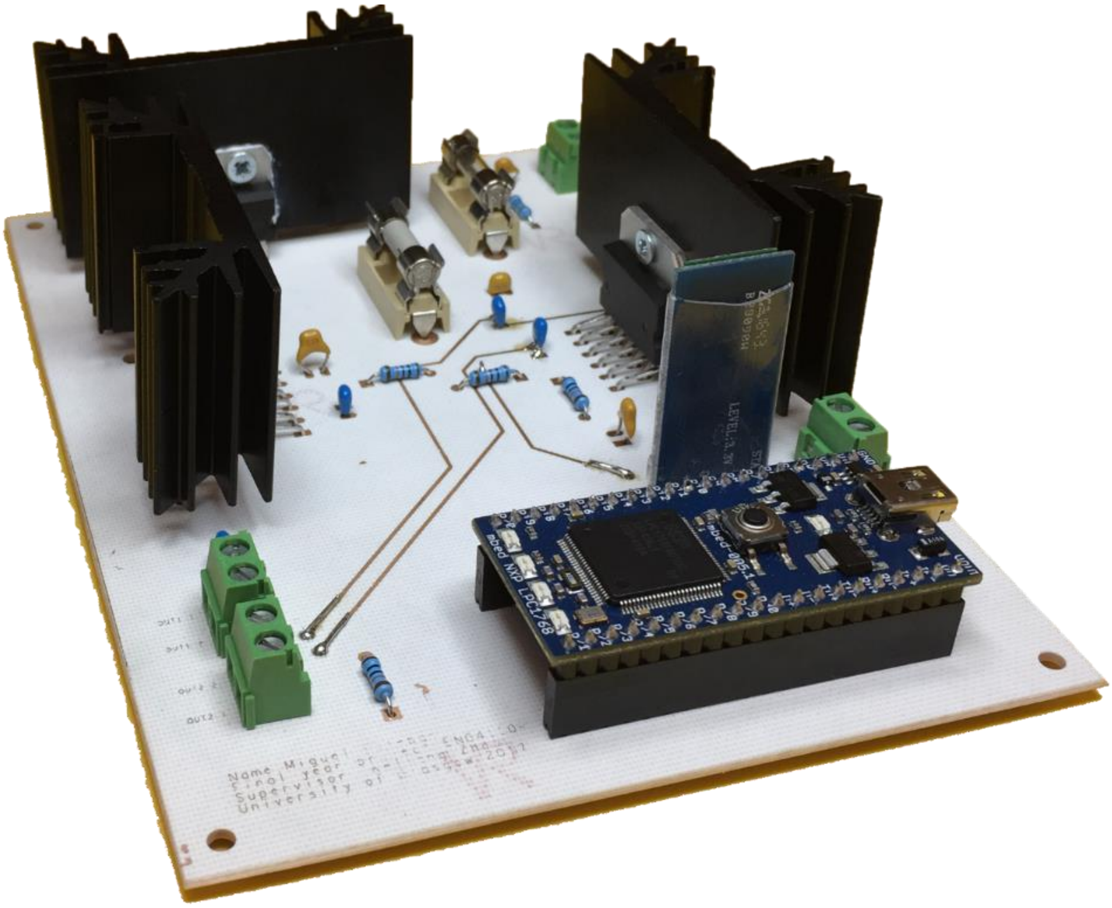


Figure 7.Circuit Assembly

4. Sinusoidal PWM wave

Nowadays the majority of frequency converters use power electronics to generate the alternating wave, and the Pulse Width Modulation (PWM) technique is the base for generating the output signal.

Pulse Width Modulation allows us getting analogue results driven by digital devices. The operating principle is based on a logical voltage pulled out from a microcontroller to the gate of a MOSFET (or the base if it is BJT) changing the state of the transistor and making it act like a switch. This control creates a square wave of a fixed voltage. The fast-acting switches in the H-Bridge produces voltage pulses at a constant magnitude proportional to the DC bus voltage.[24]

The objective of an inverter circuit is to generate an alternating current the closer to a sinusoidal shape. The output voltage using PWM technique has only two possible values: the voltage of the DC bus that supplies it or zero. Therefore, the sinusoidal wave is going to be shaped by combining different square waves in small intervals of the same amplitude and frequency but different duty-cycle. This technique is what is called Sinusoidal PWM or SPWM.

4.1 Unipolar SPWM

The switches are grouped in pairs, as it was mentioned in Section 3.1; one pair will generate the positive component and the other generates the negative component of the sine wave. As we can see in the first plot of Figure 8, the resultant output voltage is higher when the longer the switch is on (or lower in case the switches that generate the negative component are on). Thus, making variable this width of pulse, we can get different resultant voltage output values, and if the switching is faster enough, we can generate a discrete sinusoidal wave. In addition, the relation between those intervals duration determine the output frequency of the sine wave, which is what we expect to control.[15]

Regarding the DC-AC converters theory, the control of the switches is conducted by the comparison of two analogue signals, a carrier signal and a control signal. In Figure 8 first plot the carrier signal is the triangular wave (v_{tri}) and the modulating or control signal is the sinusoidal ($v_{control}$). This one will define the output wave frequency, while the carrier wave will set the speed at which those switches turned on and off. If the frequency of this carrier signal is increased, the associated output can have much higher resolution.[15], [25] The technique that is going to be implemented is Unipolar PWM, where two control signals instead of one (one positive and one negative), are compared with a triangular carrier signal. In PWM with unipolar voltage, the switches in the two legs of the H-bridge are not switched simultaneously. Taking over the naming of the switches of Figure 3.b, the legs A and B are controlled separately, each one controlled by the comparison of v_{tri} with $v_{control}$ and $-v_{control}$, respectively. In Figure 8 it is shown in the second plot the resultant wave of leg A. If the $v_{control}$ is greater than v_{tri}, then transistor A+ will be activated and therefore the voltage will be high; and consequently A- will be activated when the opposite. Leg B will be compared with the negative control signal, as it is shown in the third plot. The resultant output wave will be the difference between leg A output and leg B output, and the result is shown in the last plot of Figure 8. The final output wave is square wave with a variation in the pulses width, and as it is represented in dash line, the fundamental harmonic of it is a sine wave.[25][26]

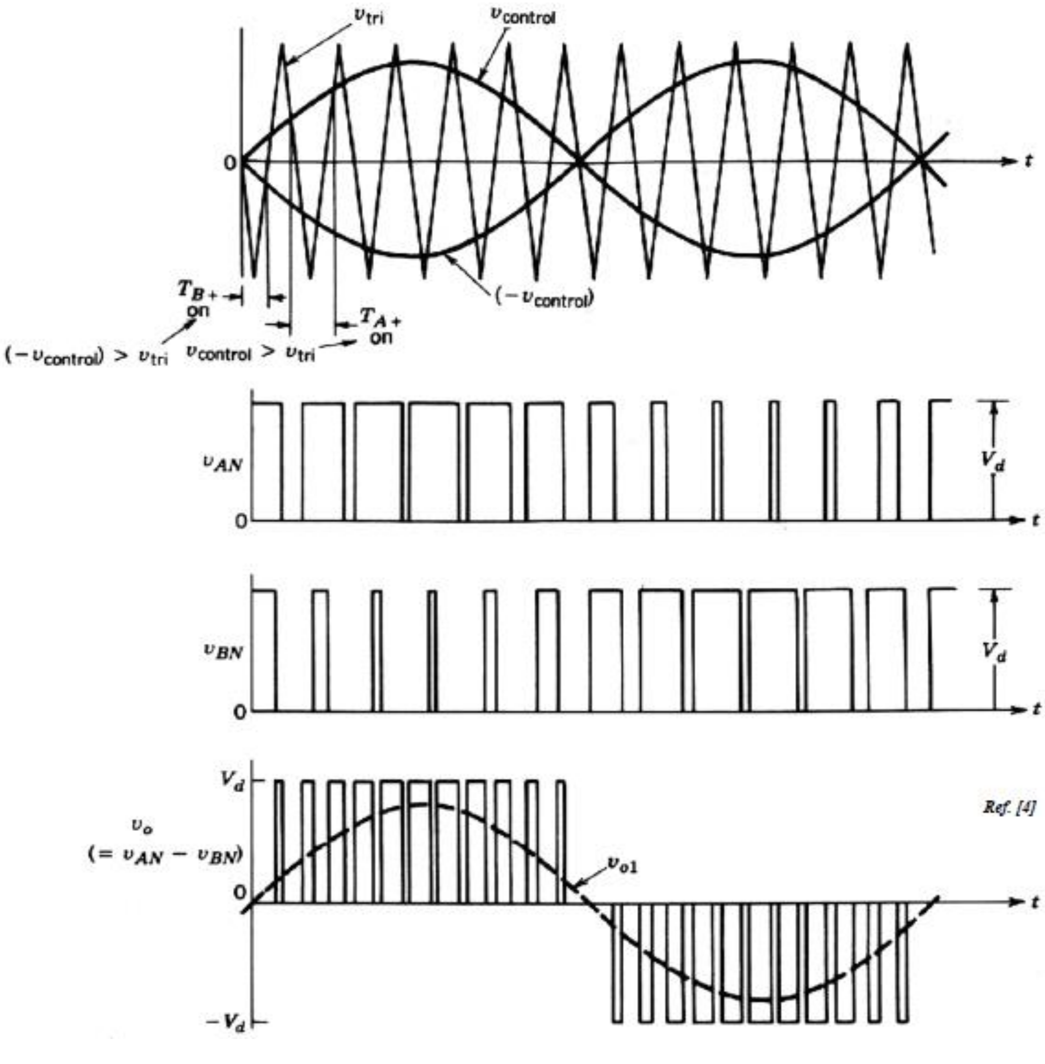


Figure 8.Unipolar Sinusoidal PWM [27]

This technique, as it compares two control signals, doubles the switching frequency of the transistors with regard to a conventional PWM technique. The advantage on this appears in the harmonic spectrum of the output voltage waveform, where the lowest harmonics appear as sidebands of twice the switching frequency. Furthermore, if the carrier frequency is increased, the final output has better resolution, resulting in a smoother output waveform with less ripple/distortion. This enhances the motor torque performance and decreases audible motor lamination noise. Faster switching is always better for output controllability, but it involves other kind of constrains harmful for the transistors live that will be commented in the next section (Section 4.2). [15], [25]

4.2 Digital Implementation

In the previous section was defined the working principle of the SPWM technique using the comparison of two analogue signals to generate the output wave. However, the control signals to the transistor are going to be generated from the selected Mbed NXP LPC1768 microcontroller (Figure 5.a), which means no analogue signals are compared. The Mbed microcontroller will generate a digital signal with an output voltage either 0 or 5 V.

Mbed library has a PWM function *PwmOut* for an established duty-cycle value. To generate the desired waveform, the duty-cycle must change its value several times along a whole period. This introduces two matters: which are those values and when must they change.

To calculate the values needed we can regard Figure 8 last plot and observe the variation of the pulse width in each stage of the sine wave. To implement those discrete interval values that together generate a fundamental harmonic of a sine, first we need to divide the whole period into a selected number of intervals. The higher the number of intervals the more switching of the transistors will be required and, consequently, the smoother will be the resultant wave form. Just as an example, we can take a reduced number of intervals or steps to easily visualise the implantation concept. For instance, the sine period will be divided into 10 steps, and their corresponding values will be calculated with equation (4.1).

$$Sine\ Duty\ (i) = sine\left(\frac{2\pi}{N_{steps}} * i\right)\ [rad] \qquad (4.1)$$

The values are stored in an array of N_{steps} components. The result will be an array with positive and negative values between 0 and 1; positive values will be sent to the A leg of transistors and negative values to the B leg. The *PwmOut* can only admit values between 0 and 1, so we need to use the absolute value of them. As the absolute values are symmetrical, the first half of them (positive values) will activate the first leg of the H-Bridge and the rest (negative values) the second one. With the 10 steps example the values obtained are represented in Table 1.

Step [i]	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Sine duty[i]	0.588	0.951	0.951	0.588	0.00	0.588	0.951	0.951	0.588	0.00

Table 1.Sine duty values for 10 intervals

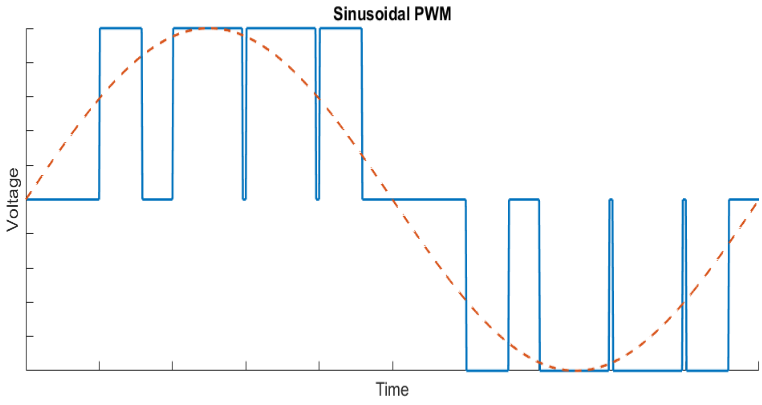
The second matter involves the frequency at which these values must change. The value will be updated every period of this updating frequency using an *Interrupt Ticker*, calculated according equation (4.2).

$$F_{update} = \frac{F_{sine\ wave}}{N_{steps}} \quad (4.2)$$

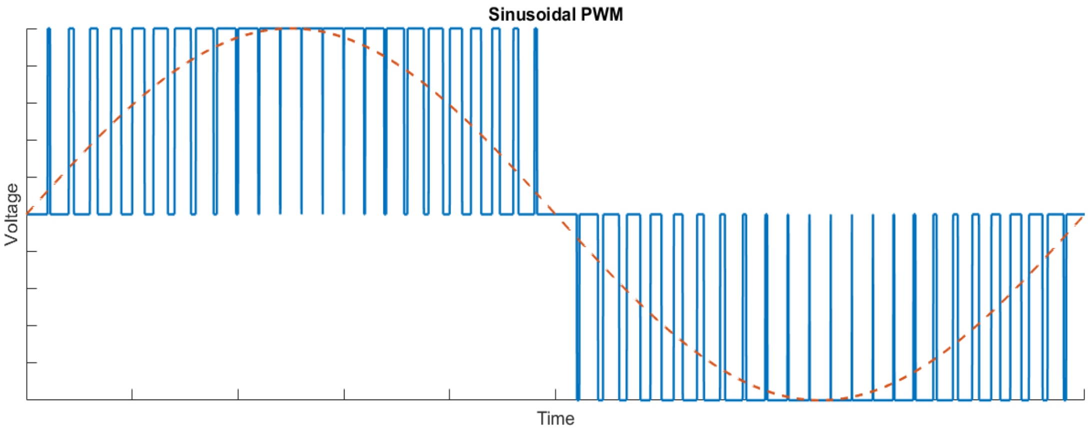
The microcontroller will send three output signals per phase: *PwmOut*, IN1 and IN2. These both signals select which transistors are activated in each moment. When the PwmOut is pulling out the duty-cycle value pertaining to an interval in the first half, then IN1 is set high and IN2 is set low, and the other way round when the interval belongs to the second half. The interrupt will call a subroutine in the code that updates the value of the PWM and the values of the IN1 and IN2 signals. It is important that IN1 and IN2 are never set high at the same time because a short-circuit of the power source would occur.

Remaining the 10 steps example, in Figure 9.a is represented a simulation using Matlab of the final output waveform. If we increment the number of steps, which means rising the carrier frequency explained in the previous section, we can get an enhancement of the wave. The final number of steps is 50 steps, because it offers a good enough performance (Figure 9.b). We could increment the number of steps but that is translated into rising the frequency of switching of the transistor, which introduces problems with heat dissipation and the risk of failure of one of them increments.

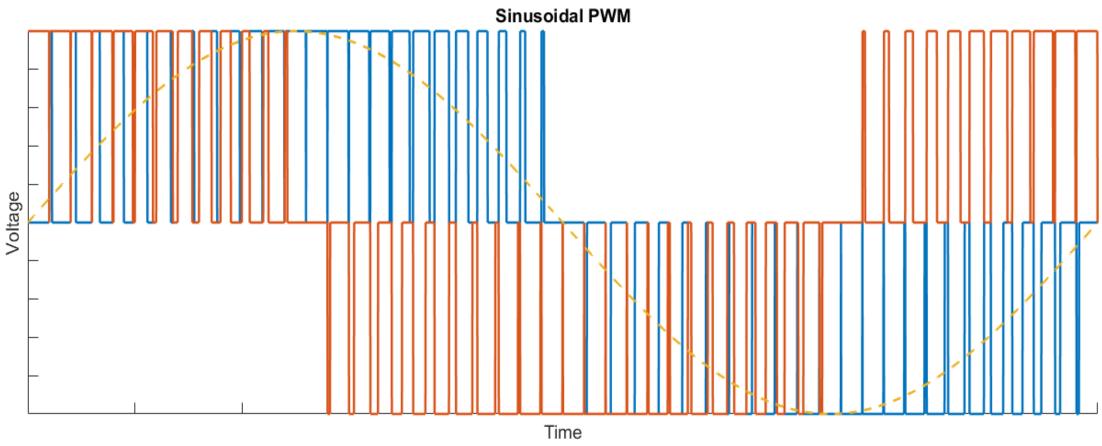
This far the implementation of the driving signals of one phase has been explained, but it is necessary to include the second phase. The phases have a delay of 90^{o} between each other, what can be translated into a delay of the number of steps divided by four. Thus, to pull out the signals of the two phases with the corresponding delay at the same time, the interval index of the second one must start in the step number $N_{steps}/4$ (Figure 9.c).



(a)



(b)



(c)

Figure 9.Simulation of SPWM in Matlab. (a) 10 steps. (b) 50 steps. (c) 50 steps two-phases

5. Software implementation

Today, Android based mobile phones are available with large number of people around the world. Bluetooth operates over the unlicensed, globally available frequency of 2.4GHz. It can link digital devices within the range of about 100m. [28]

The Mbed microcontroller is programmed through the online platform Developer Mbed Workspace.[29] The code programmed has the aim to generate the 6 output signals through different pins needed to drive the H-Bridges. The states of frequency and direction of rotation are controlled via an Android app that communicates via Bluetooth Serial Connexion with the microcontroller. The development and operation of the user interface in Android platform is further addressed in Chapter 7, and the Bluetooth communication is explained in Section 5.2.

5.1 Mbed program

During the development of the project, multiple versions of the code have been written to test the power stage, depending on the way of controlling it (fixed frequency, push button, Matlab interface or Android interface) before all the communications were set up. The code of the final Mbed program to control with the smart phone app is written in APPENDIX C. The explanation of the code throughout this section relates to the flow chart diagram drawn in Figure 10. There are three different parts in the code: Main Program, Interrupt subroutine and *PwmOut* function.

The Main Program first sets up the values of the constants, the function of the utilized pins, initializes the variables and defines the communication settings details such as baud rate. In this section the frequency of the PWM and the number of steps chosen are defined and will remain constant during the entire program. With the chosen steps now, we can calculate all the possible values the duty cycle of PWM can adopt, and they are stored in an array. What has been defined until this point is executed once and the rest of the main program is executed inside an infinite loop, repeating the same process. Inside this loop, the program waits for the Bluetooth to be readable; it means it has new data to process. The data sent by the Bluetooth module is one-byte length. Depending on the number arrived, the state of rotation changes, it means the variables read by the Interrupt subroutine change their value.

The different states are:

- Start: The motor starts rotating forward by activating the transistors.

- Stop: The motor changes the direction of the current for a few seconds to make the motor stop and then it turns of the switches.
- Rotate forward: Changes to direction of rotation to forward if it was reverse.
- Rotate reverse: Changes to direction of rotation to reverse if it was forward.
- Change frequency: It changes the current frequency to the last value got from the Bluetooth module, changing too the period of the ticker interrupt

The Interrupt subroutine occurs at a defined period using a ticker depending on the actual value of the output frequency. It uses the values of the variables given in the main program to check if the current state has changed. It updates those values (stop, start forward or start reverse) and the duty-cycle array indexes of each phase. Then it sets the pins 11, 12, 13 and 14 of the microcontroller according to those indexes and variables. These pins output the signals IN1.1, IN1.2, IN2.1 and IN2.2, which able or disable the pairs of transistors in the H-Bridge. Then, once the pins have been set, the subroutine updates the two *PwmOut* signals with the new duty cycle values.

The *PwmOut* is a predefined function from Mbed library that generates a PWM signal of a fixed duty cycle. This function outputs two different PWM signals through pins 23 and 22 of the microcontroller that will be connected to the ENABLE pins of each H-Bridge. The value of the duty cycle needs to be updated, and that will be carried out by the Interrupt subroutine as mentioned above.

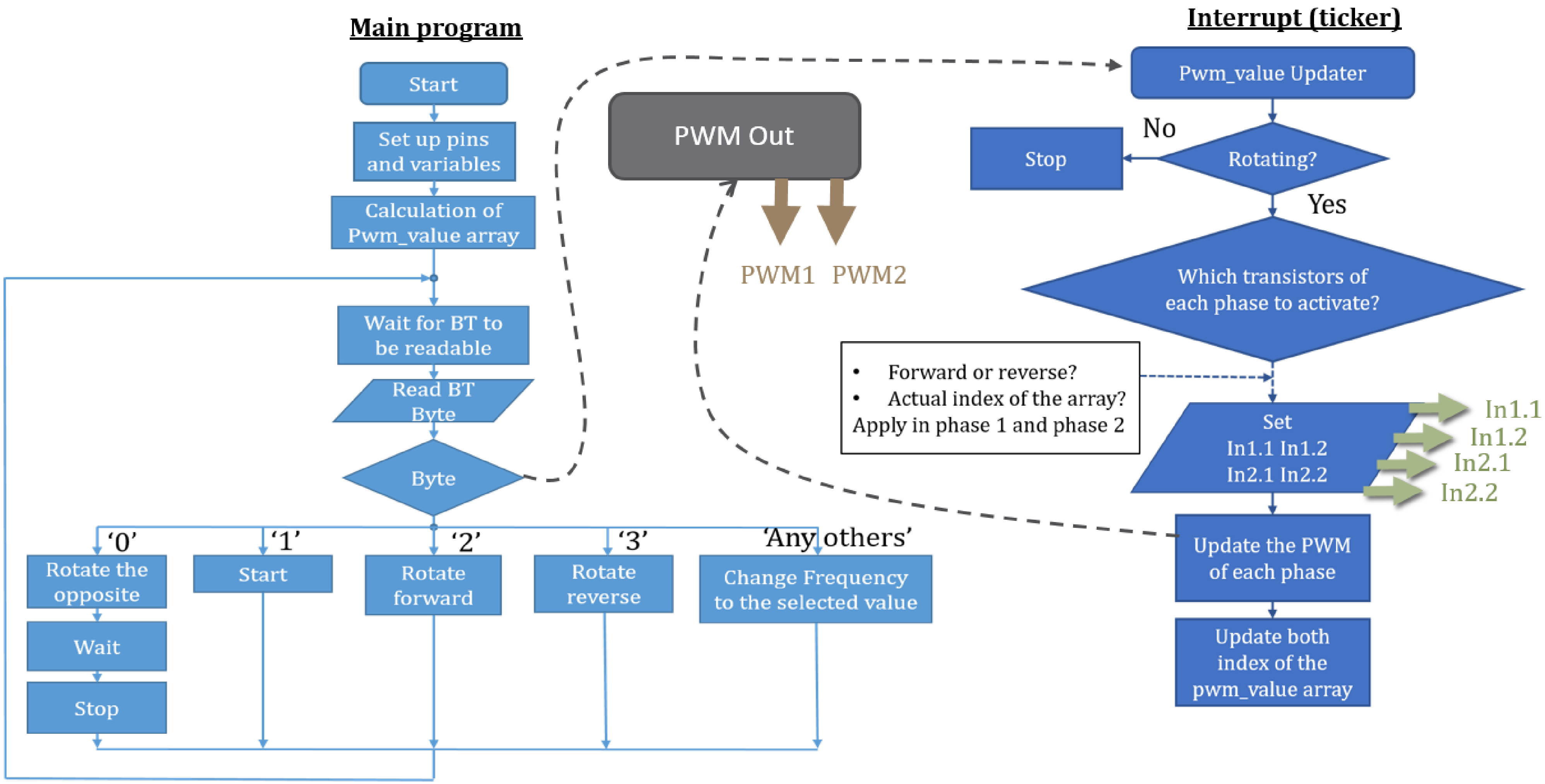


Figure 10.Flow Chart

5.2 Bluetooth serial communication

Serial communication is a protocol used to send and receive information and data between computers or electronic modules. It is a simple protocol of communication but currently used in many applications due to its easy implementation and reliability in applications that do not require big quantities of data sent at the same time. The Serial link has one unidirectional channel for sending data and another one for receiving. Both devices connected must have the same configuration settings because the serial link is an asynchronous link. The Bluetooth dispositive is Bluetooth Module HC-05 and it is the link of serial communication between the smart phone app and the microcontroller. It will use a serial communication via electrical connexion with the microcontroller and a wireless Bluetooth connexion with the Android device. [30]

The Mbed microcontroller has two serial connections, one through the USB port, to connect with a computer, and other two using two pins for transmissions. The one going to be used uses pin 9 to transmit data (TX) and pin 10 to receive data (RX). Pin 9 (TX) will be connected to pin RX of the Bluetooth module and pin 10 (RX) to pin TX.

As it uses asynchronous communication, Serial channels have parameters to configure in both devices that are communicating:

- Baud Rate: It defines the number of bits per second that it is going to be transmitted. There are some standard baud rate ranges typically used, the default setting is 9600.
- Data length: The number of bits transferred, typically 8 bits (1 byte).
- Parity: It is an optional bit that can be added to make sure the transmission is correct, but it is not going to be used in the communication, so it will be set “None”.
- Stop Bits: To end the transmission either one or two bits are added. Default is one.

The configuration of the HC-05 Module can be set up (or just checked out) via pc connecting it to the microcontroller and this one to the pc through the USB port. In the datasheet it includes a list of commands to change the communication settings when connected to different microcontrollers. In this communication, the default settings are used but it was still necessary to check which these default parameters were.

The communication setting used between Mbed microcontroller and Bluetooth module are a baud rate of 115200 baud and a transmission: 8-N-1. This is the common notation for the transmission of 8 bits, none parity bit and one stop bit. (Figure 11)

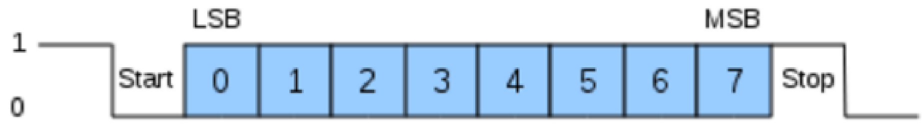


Figure 11.Asynchronous 8-N-1 transmission.

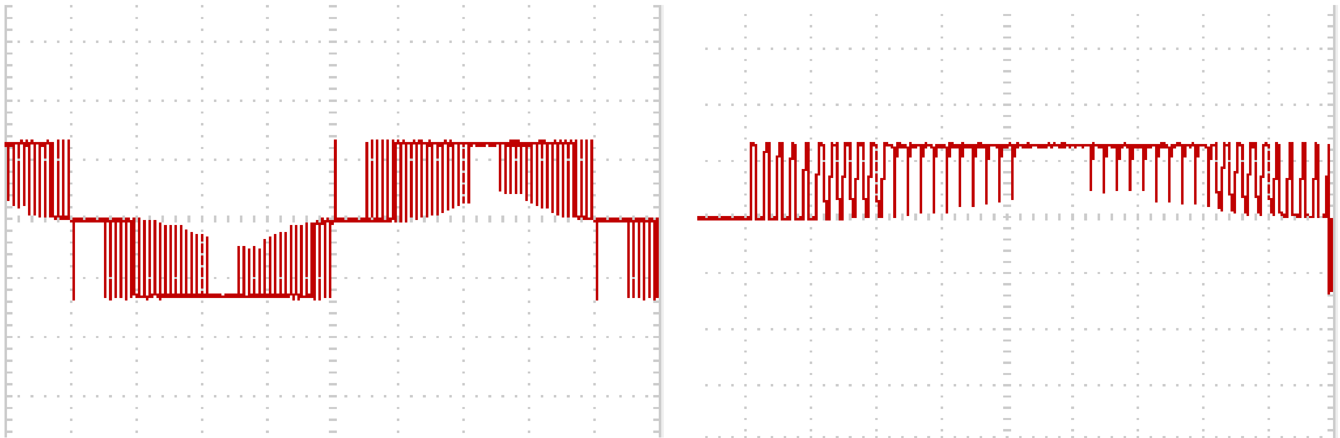
The wireless communication with the Android app is going to be transmitted at 9600 baud rate and 8-N-1 transmission too. The Android app is programmed in *Android Studio* platform with this settings too. The code is included in APPENDIX D.

6. Circuit behaviour

Once the power stage is assembled (first only a provisional circuit and later the PCB), it is time to look for results of its performance. First, to protect the circuit devices it is going to be tested at some fixed frequencies using a simple resistor load at a reduced voltage, and then once it has been checked the output waveform is the expected, it will be tested with the Irn Bru motor load at 30 V.

6.1 Resistor load

A resistor of 10kΩ is connected between the output pins of each phase in the power stage. The voltage is supplied by a variable power source, setting it a 12 V and current limiting the power stage at 0.5 A to protect it. The oscilloscope channels are connected to the terminals of the resistor of one phase, and in the oscilloscope we plot the difference between them. The waveform resulted is the one shown in Figure 12.a for 50Hz current frequency and Figure 12.b for 20Hz. The number of steps per period has been reduced and set up to 20 steps to enhance and make clearer the oscilloscope plot.



(a) *(b)*

Figure 12. Oscilloscope screenshot for resistor load. (a) 50Hz current. (b) 20Hz current.

It can be affirmed that the waveform is similar to the one expected, and the current consumed is just a few of mA (around 20 mA); thus, the motor can be tested.

6.2 Motor load

The Irn Bru induction motor is connected to the power stage supplying a voltage of 30 V DC from the variable source and current limiting it at 1 A. It rotates at different frequencies and both forward and reverse directions with a total consumption of 0.76 A. In the 'Stop' state of the program, the variable source supplies 0.22A that are consumed by the sum of the microcontroller, Bluetooth module and LED.

Analysing these results, the conclusions are that the variable power source currently used can be replaced for a smaller fixed power supply. The one selected is from manufacturer RS model 30 V DC 1 Output; Right Angle Jack Switched Mode Power Supply, 1A; contributing with that in the project objectives of making the stage more portable.

In addition, it is necessary to include heat sinks to dissipate the heat generated on the integrated circuits, such as the two H-Bridge and especially on the voltage regulator, which is heated to high temperature. The heat sink allocated is one model recommended by the laboratory technicians because it dissipates enough heat and is the one they have in the store. [22]

7. User interface

User interfaces make possible the interaction between humans and machines, with the goal of allowing effective operation of the machine from the human end. Tangible User Interfaces are those in which the user interacts with a digital system manipulating physical objects. As it is said in [31]: "They started with the invention of mouse and keyboard; and since the last two decades they have experienced tremendous growth and has provided us with a link between physical and digital world."

Nowadays, to perform their daily activities over 2 billion users rely on smartphones and tablets. Not only do users play games or send messages, they use mobile applications (a.k.a., apps) for every type of need, including social and emergency connectivity.[32]

Android platform supports the majority of smartphones operating systems and it has the advantage of being easy and free to program and to install your own app in it, without the necessity of special licenses. Demonstrating the Android app functionality to select the state of rotation of the motor during the Open Day will show an innovative facet and up-to-date image of the University.

The app is going to be created using Android Studio software. "Android Studio provides the fastest tools for creating apps on all kinds of Android devices. First-rate code editing, debugging, performance tools, a flexible compilation system, and an instant compilation and implementation system allow you to focus on creating unique, high-quality applications." [33]

7.1 Android user interface

Android platform uses Java language to execute its apps. In Android Studio there are included some Android libraries compatible with Java language making the code more intuitive and approaching the creation of a basic app to anyone that holds any sort of programming skills. In Android Developer site there can be found several examples of the use of multiple function or procedure that may be interesting to include in an app, such as a predefined code for setting any Bluetooth connexion.

The Android app is programmed in some blocks. Manifest file declares the app components, libraries, hardware and software required for the app and identifies the user permissions necessary for running the app. In this app, only Bluetooth permission is necessary, and the Manifest file code can be read in APPENDIX C.

The app components are the essential building blocks of an Android app, which are an entry point through which the system or a user can enter your app. In the development, only one Activity component is going to be used, which is divided in two sections: the Relative Layout, where the appearance of the app is programmed; and the Main Activity, which contains the functionality of the elements defined in the Layout.

The Relative Layout can be programmed writing the code or building it with graphic tools, making it more intuitive, and code will automatically generated. The Layout code automatically generated can be found in APPENDIX C and the graphic Layout in Figure 13.

Figure 13.App Layout

The background colours represent the Irn Bru brand and it has three different touch interactive buttons. The first one is a Switch Button with an “ON/OFF” label beside it indicating it function.

In the middle of the background, we can find a Radio Group with two Radio Buttons (only one can be pressed at the same time) that indicate if the motor is rotating forward or reverse. Beneath it a Seek Bar and its respective label indicates the frequency of the current that drives the motor. The Text Toast appears at the bottom down the app when a message is sent to inform the user.

The Main Activity code (APPENDIX C) is the core code of the app and it assigns the functions to the elements defined in the Relative Layout. In the main activity, also the Bluetooth connexion is set at the starting launch of the app. As commented before, there are three buttons defined in the Layout, and when any of them is interacted by the user, an interrupt subroutine occurs.

The Switch Button can be either ON or OFF, and when it changes it will check the actual state, will send to the microcontroller the pertinent byte and a Text Toast will appear indicating “Start” or “Stop”.

The Radio Group ,which selects either rotate forward or reverse, works in a similar way than the Switch Button, but instead of checking the state of the button, as there are more than one, it resorts to check the element ID of the button to find out which one is hold. Only one of the two buttons can be hold at the same time, and when the button hold is changed, a message in the Text Toast will appear to inform "Changing to rotate forward" or "Changing to rotate reverse".

Finally, the Seek Bar is employed to select the frequency of the magnetic field in the rotor in a range from 20 Hz to 60 Hz. The current frequency of rotation is indicated in a label above the Seek Bar that says: “Frequency: N Hz”, being N the frequency selected. When touched, it introduces three different interrupt subroutines: one *On Start* pressing, one *On Progress* and one *On Stop*. *On Start* and *On Progress* it would do nothing but displaying the frequency selected at every instant. It would be *On Stop* when apart from displaying the frequency; it will send it to the microcontroller to be changed on the motor.

8. Conclusions and further work

The objectives exposed in the Introduction of the project (Section 1.1) have been totally achieved. The size of the power stage have been reduced to fix inside the motor box, and even the power source have been replaced for a small fixed one. In Figure 14 it is shown the actual motor devices, and we can compare them with the ones in APPENDIX A to observe the size reduction. Now the power stage is more portable to carry to the Open Day demonstration.

Figure 14. Final devices.

The different states of rotation at different frequencies have also been achieved, having the capacity of selecting a range of frequencies of supply from 20 to 60 Hz in both directions, forward and reverse. In addition, with the implementation of a user interface in Android platform and controlling the frequency through Bluetooth wireless connexion, the Open Day Demonstrator becomes more attractive to the audience, showing an innovative perspective of the degree without losing the essence of the Induction Motor and the representation of Scotland through the Irn Bru soft drink.

Despite the objectives of the project have been achieved, due to the time constrains that a Final Year Project has, there are some add-ons or improvements that can be the result of further works on the project.

The designed stage outputs the selected current frequency, but it has not any feedback of the real speed of the motor. A speed sensor could be allocated to measure the rotor speed and send the response to the microcontroller back. To control the desired speed, a controller could be designed using Matlab and then implementing the code in the Mbed microcontroller. Thus, the seek bar in the Android app will control real speed instead of frequency of the sinusoidal wave that drives the motor.

In the Open Day, Mr. Calum Cossar uses a quiz programmed in Python related to Renewable Energies in Scotland. When the user guess the right answers the motor starts rotating. This quiz could be programmed in the Android app to avoid using the computer and give the user a tablet to answer it, making it more portable and accessible.

The accomplishment of the project has had significant impact on me gaining experience with power electronics, C programming and learning Java programming for Android apps. The opportunity of carrying out this project in a foreign university made me acquire awareness of different working methods and improved my English reading comprehension, vocabulary and writing skills.

References

[1] "Motores de inducción alimentados por convertidores de frecuencia PWM," *Mot. Distrib.*, vol. 23, no. 12, pp. 1–3636, 2009.

[2] A. Bucher *et al.*, "Design of a Full SiC Voltage Source Inverter for Electric Vehicle Applications Siemens AG , eCar Powertrain Systems Keywords," *18th Eur. Conf. Power Electron. Appl.*, pp. 1–10, 2016.

[3] P. K. Sahu, P. Shaw, and S. Maity, "4- Modeling and Control of Grid-Connected DC / AC Converters for Single-Phase Micro-inverter Application," *Ieee*, 2015.

[4] P. T. Krein and R. S. Balog, "Cost-Effective hundred-year life for single-phase inverters and rectifiers in solar and LED lighting applications based on minimum capacitance requirements and a ripple power port," *Conf. Proc. - IEEE Appl. Power Electron. Conf. Expo. - APEC*, pp. 620–625, 2009.

[5] T. Nishida, S. Moisseev, E. Hiraki, and M. Nakaoka, "Duty Cycle Controlled Soft Commutation High Frequency Inverter for Consumer Induction Cooker and Steamer," *IECON Proc. (Industrial Electron. Conf.*, vol. 2, pp. 1846–1851, 2003.

[6] "IEC 60050 - International Electrotechnical Vocabulary - Details for IEV number 411-31-10: "induction machine""

[7] "Electropaedia, Energy Sources and Energy Storage, Battery and Energy Encyclopaedia and History of Technology." [Online]. Available: http://mpoweruk.com/. [Accessed: 02-Apr-2017].

[8] T. J. Blalock, "The first polyphase system a look back at two-phase power for AC distribution," *IEEE Power Energy Mag.*, vol. 2, no. 2, pp. 63–66, Mar. 2004.

[9] T. Croft and W. Summers, "American Electricans' Handbook," Eleventh E., New York: McGraw Hill, 1987, p. page 3–10, figure 3–23.

[10] T. Audel, "Hawkings electrical guide. Figure 1253," in *Hawkings electrical guide*, New York, 1917, p. 3367.

[11] F. Blaabjerg, F. Lungeanu, K. Skaug, and M. Tonnes, "Two-phase induction motor drives: Low-cost topologies for TPIM drives in industrial applications," *IEEE Ind. Appl.*

Mag., vol. 10, no. 4, pp. 24–32, 2004.

[12] N. Mohan, T. M. Undeland, and W. P. Robbins, “Chapter 5. Line-Frequency Diode Rectifiers,” in *Power Electronics*, 2003.

[13] N. Mohan, T. M. Undeland, and W. P. Robbins, “Figure 7-27.Full Bridge converter,” in *Power Electronics*, 2003.

[14] N. Mohan, T. M. Undeland, and W. P. Robbins, “Chapter 2. Overview of Power Semiconductor Switches,” in *Power Electronics*, 2003.

[15] “What is a frequency inverter?” [Online]. Available: http://www.frequencyinverters.org/what-is-a-frequency-inverter-106779.html. [Accessed: 03-Apr-2017].

[16] N. Mohan, T. M. Undeland, and W. P. Robbins, “Section 7-7. Full Bridge DC-DC Converters,” in *Power Electronics*, 2003.

[17] “H-Bridges – the Basics | Modular Circuits.” [Online]. Available: http://www.modularcircuits.com/blog/articles/h-bridge-secrets/h-bridges-the-basics/. [Accessed: 03-Apr-2017].

[18] C. Blake and C. Bull, “IGBT or MOSFET : Choose Wisely,” *Source*, 1970.

[19] D. Full and B. Driver, “L6201 l6202 - l6203,” *Current*, no. July, pp. 1–20, 1997.

[20] N. X. P. Semiconductors, “LPC1769/68/67/66/65/64/63 32-bit ARM Cortex-M3 microcontroller; up to 512 kB flash and 64 kB SRAM with Ethernet, USB 2.0 Host/Device/OTG, CAN,” *Quadrature*, no. November, 2011.

[21] St, “L78xx SERIES Positive Voltage Regulators,” no. November, pp. 1–34, 2004.

[22] A. Phillips, “Laboratory technician.”

[23] B. M. Score, “Ceramic or electrolytic output capacitors in DC / DC converters — Why not both ?,” 2015.

[24] “PWM Arduino.” [Online]. Available: https://www.arduino.cc/en/Tutorial/PWM. [Accessed: 05-Apr-2017].

[25] N. Mohan, T. M. Undeland, and W. P. Robbins, “Chapter 8. Switch-Mode dc-ac Inverters: dc-sinusoidal ac,” in *Power Electronics*, 2003.

[26] T. C. C. Ca-ca, “Tema 6 . Convertidores CC-CA y CA-CA,” pp. 1–22, 2015.

[27] N. Mohan, T. M. Undeland, and W. P. Robbins, “Figure 8-15. PWM with unipolar voltage switching.,” in *Power Electronics*, 2003.

[28] D. Sunehra and V. M., “Implementation of Interactive Home Automation Systems Based on Email and Bluetooth Technologies,” *IEEE Electron. Commun. Eng.*, vol. 7, pp. 458–463, 2015.

[29] “mbed Compiler Workspace Management.” [Online]. Available: https://developer.mbed.org/compiler/#nav:/; [Accessed: 04-Apr-2017].

[30] “Homepage - Handbook | mbed.” [Online]. Available: https://developer.mbed.org/handbook/Homepage. [Accessed: 04-Apr-2017].

[31] V. Kumar, T. Choudhury, and S. Sabitha, “Physical Telepresence : Growth trends of Tangible User Interface and its future,” no. October, pp. 229–235, 2016.

[32] D. Di Nucci, F. Palomba, A. Prota, A. Panichella, A. Zaidman, and A. De Lucia, “Software-based energy profiling of Android apps: Simple, efficient and reliable?,” *2017 IEEE 24th Int. Conf. Softw. Anal. Evol. Reengineering*, pp. 103–114, 2017.

[33] “Develop Apps | Android Developers.” [Online]. Available: https://developer.android.com. [Accessed: 06-Apr-2017].

APPENDICES

APPENDIX A. Former devices

In this appendix are included pictures of the devices used to drive the motor before the redesign of the power stage and their dimensions.

Figure A. 1. Power source. 500x250x200 mm

Figure A. 2. Controller. 400x350x100mm

Figure A. 3.One phase Inverter. 300x400x250mm

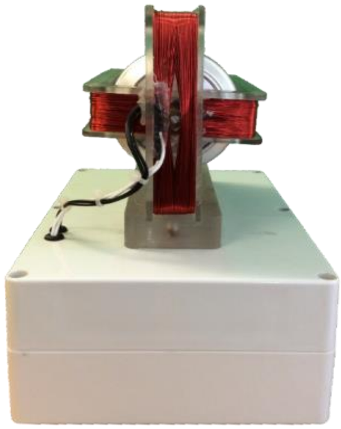


Figure A. 4. Irn Bru Motor. 250x150x300mm

APPENDIX B. Schematic and PCB files.

This appendix contains the schematic files of power stage along with the PCB production files. Apart from graphically clarifying the power stage design with the schematic file, this appendix has also the aim of making available the files for anyone interested on printing the PCB.

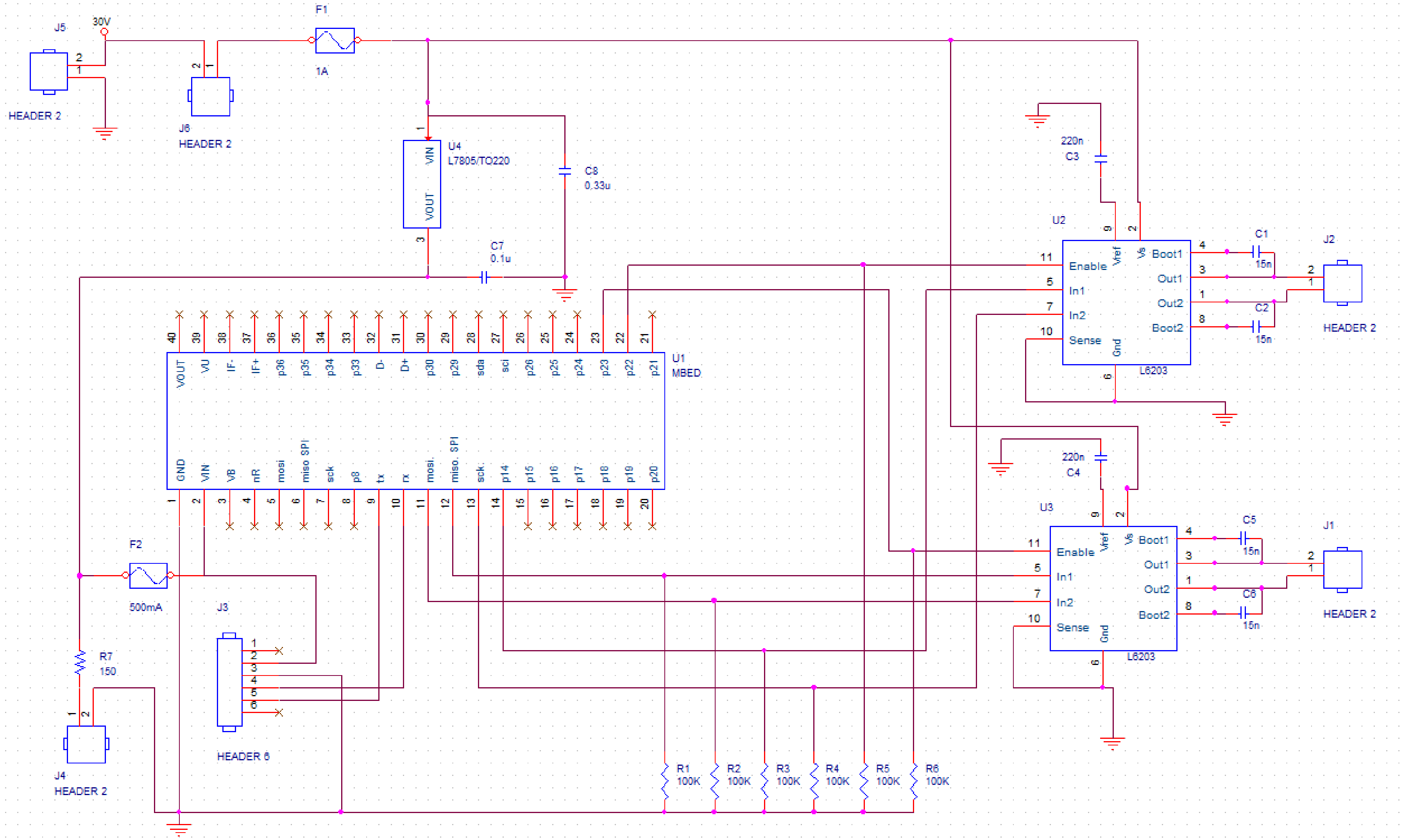


Figure B. 1. Schematic of the power stage.

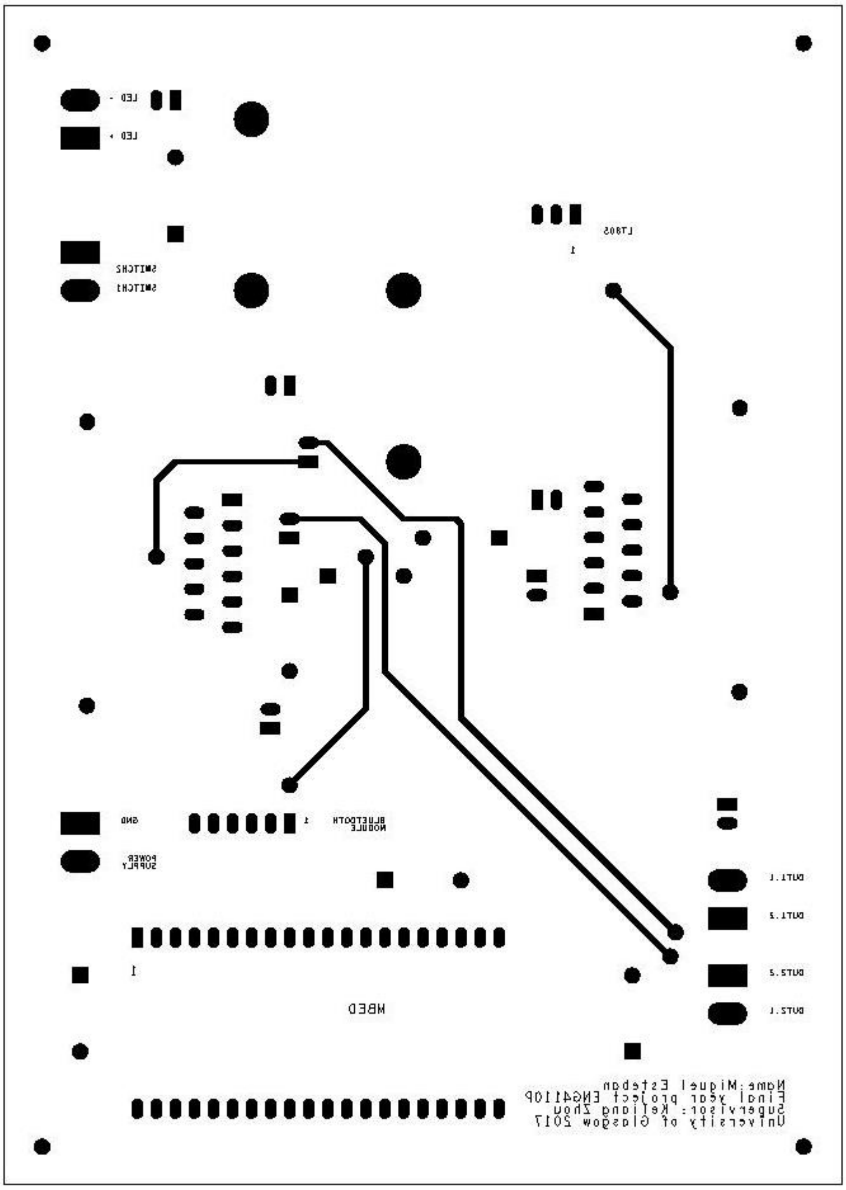


Figure B. 2. PCB top layer production file.

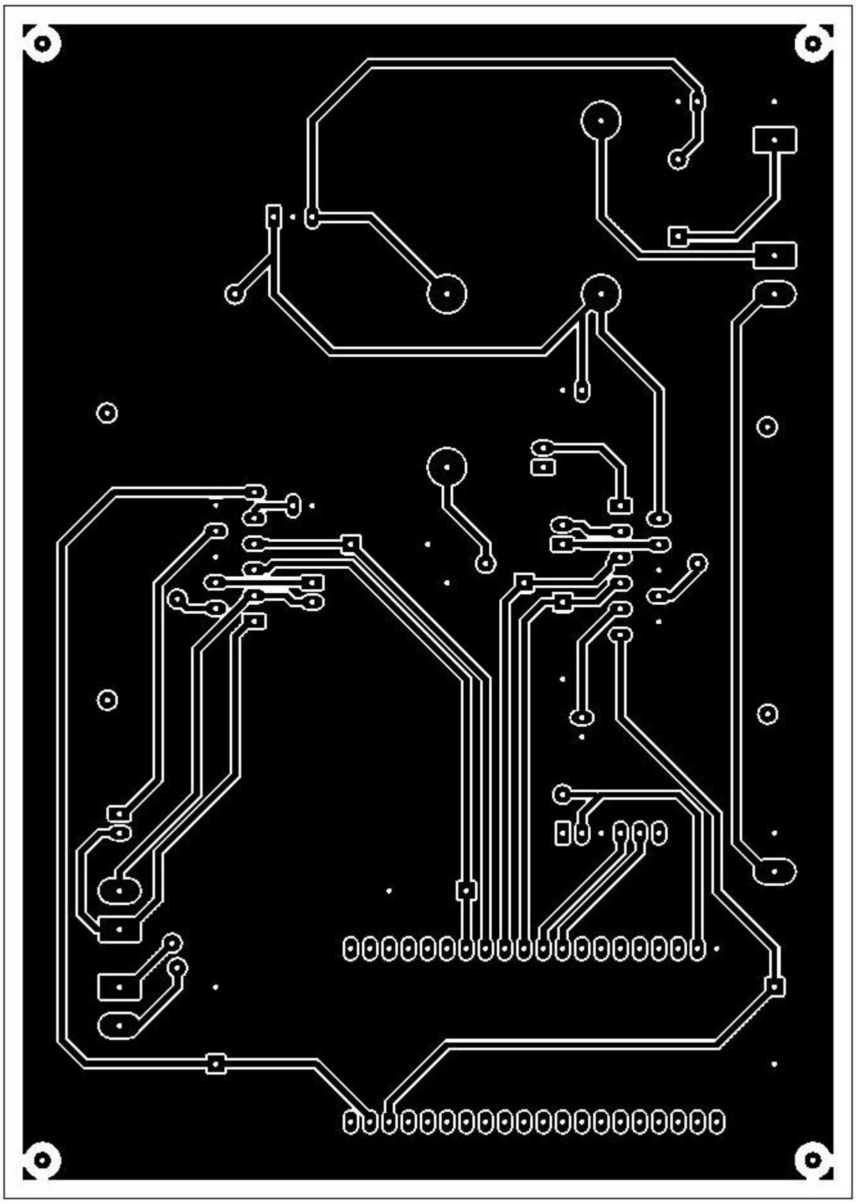


Figure B. 3. PCB bottom layer production file.

APPENDIX C. Mbed Microcontroller Code.

```
#include "mbed.h"
//CONSTANTS
#define PI                  3.141592f
#define SINE_DIVISION       50 //Constant value to create the array
pwm_value

//SET UP VARIABLES
int FREQ_PWM=5000;//5kHz
float pwm_value[SINE_DIVISION];//Possible values of PWM
//Initial Standard Values
int SINE_FREQ=20;
bool reverse=0;//0 rotates forward; 1 rotates reverse
bool start=0;//0 stops the motor; 1 rotates
int idx1=0;//Index for the array component of phase 1
int idx2=SINE_DIVISION/4;//Index for phase 2; 90 degrees delay

//SET UP PINS
DigitalOut in11(p11);//Enable transistors A+ y B- of phase 1
DigitalOut in12(p12);//Enable transistors A- y B+ of phase 1
DigitalOut in21(p13);//Enable transistors A+ y B- of phase 2
DigitalOut in22(p14);//Enable transistors A- y B+ of phase 2
PwmOut spwm_phase1 (p22);//Output PWM signal of phase 1
PwmOut spwm_phase2 (p23);//Output PWM signal of phase 1
Serial blue(p9, p10);// Bluetooth Serial Connexion TX, RX

//TICKER INTERRUPT
Ticker change_pwm;//calls back a function to change the pwm value.
                  // It occurs at (Period of sine wave)/ (sine steps)

//INTERRUPT SUBROUTINE. PWM VALUE UPDATER
void pwm_value_updater()
{
if(start==0){// If Stop, disable transistors
        in11.write(0);
        in12.write(0);
        in21.write(0);
        in22.write(0);
}else{
if(reverse==0){         //Rotating forward
if(idx1<(SINE_DIVISION/2)){
                in11.write(1);//First half indexes
                in12.write(0);      //activate A+ & B- transistors
}else{
                in11.write(0);      // Second half indexes
                in12.write(1);      //activate A+ & B- transistors
}

if(idx2<(SINE_DIVISION/2)){
                in21.write(1);
                in22.write(0);
}else{
                in21.write(0);
                in22.write(1);
}

}else{
if(idx1<(SINE_DIVISION/2)){
```

```
                in11.write(0);//Rotating reverse --> change the input of
in12.write(1);// 1 phase to delay 180degrees is the
                                // same as swapping phases
}else{
                in11.write(1);
                in12.write(0);
}

if(idx2<(SINE_DIVISION/2)){
                in21.write(1);
                in22.write(0);
}else{
                in21.write(0);
                in22.write(1);
}
}
            //Update the new value in the PWM signals
        spwm_phase1.write(pwm_value[idx1]);
        idx1++;
        spwm_phase2.write(pwm_value[idx2]);
        idx2++;

if(idx1 == SINE_DIVISION) idx1=0;
if(idx2 == SINE_DIVISION) idx2=0;
}
}

//MAIN PROGRAM
int main()
{
    blue.baud(115200);// Set up the baud rate for BT data transmission
//Calculate the possible values according to the number of sine divisions
int i;
for(i=0; i<(SINE_DIVISION/2); i++){
        pwm_value[i]= sin(i *(2.0f* PI /(float)SINE_DIVISION));
}
for(i=(SINE_DIVISION/2); i<SINE_DIVISION; i++){
        pwm_value[i]= sin(i *(2.0f* PI /(float)SINE_DIVISION))*-1;
}
//Set up the pwm frequency
    spwm_phase1.period(1.0f/(float) FREQ_PWM);
    spwm_phase2.period(1.0f/(float) FREQ_PWM);

//INFINITE LOOP
while(1){
if(blue.readable()){//Wait for data from BT
int data= blue.getc();//Read the data (1 byte)
switch(data){
case0:                //STOP
                    reverse=! reverse;//First invert current to brake
                    wait(5);
                    start=0;//Stop
break;
case1://START
                    start=1;
      //Define the update PWM frequency according to the
 output wave period
                    change_pwm.attach(&pwm_value_updater,1.0f/
(float)(SINE_DIVISION * SINE_FREQ));
break;
case2:                        //FORWARD
```

```
                    reverse=0;
break;
case3://REVERSE
                    reverse=1;
break;
default://NEW FREQUENCY VALUE
                    SINE_FREQ= data;//Change frequency
                    change_pwm.attach(&pwm_value_updater,1.0f/
(float)(SINE_DIVISION * SINE_FREQ));
break;
}

}
}
}
```

APPENDIX D. User interface Java Code

This appendix contains the code of the Android app programmed in Android Studio; it uses Java language and Android libraries. It is divided in sections 4 sections:

Android Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="esteban.miguel.irnbrumotor">

<uses-permission android:name="android.permission.BLUETOOTH"/>
<application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
<activity android:name=".MainActivity">
<intent-filter>
<action android:name="android.intent.action.MAIN"/>

<category android:name="android.intent.category.LAUNCHER"/>
</intent-filter>
</activity><!-- ATTENTION: This was auto-generated to add Google Play
services to your project for
     App Indexing.  See https://g.co/AppIndexing/AndroidStudio for more
information. -->
<meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version"/>
</application>

</manifest>
```

Strings defined

```
<resources>
<string name="app_name">Irn Bru Prueba1</string>
<string name="action_settings">Settings</string>
<string name="ChoiceText">Choose one of the radio buttons below</string>
<string name="Forward">Forward</string>
<string name="Reverse">Reverse</string>
<string name="Start">Start</string>
<string name="DEVICE_ADDRESS">98:d3:37:00:aa:ad</string>
<string name="PORT_UUID">PORT_UUID</string>
</resources>
```

App Relative Layout

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
                xmlns:app="http://schemas.android.com/apk/res-auto"
                xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context=".MainActivity"
    tools:background="@android:color/holo_orange_dark">

<TextView
        android:text="Frequency"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="31dp"
        android:id="@+id/FrequencyText"
android:textAppearance="@style/TextAppearance.AppCompat.Headline"
        android:layout_alignBottom="@+id/skb"
android:layout_centerHorizontal="true"
        android:allowUndo="false"/>

<SeekBar
        android:layout_width="300dp"
        android:layout_height="wrap_content"
android:id="@+id/skb"
        android:max="40"
android:progress="2"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="80dp"/>

<TextView
        android:text="TextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/text"
        android:layout_alignParentTop="true"
        android:layout_alignLeft="@+id/FrequencyText"
        android:layout_alignStart="@+id/FrequencyText"
        android:layout_marginLeft="59dp"
        android:layout_marginStart="59dp"/>

<RadioGroup
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/myRadioGroup"
        android:background="#abf234"
android:checkedButton="@+id/forward"
        android:layout_marginTop="34dp"
        tools:background="@color/colorPrimary"
        android:layout_alignTop="@+id/imageView"
        android:layout_centerHorizontal="true">

<RadioButton
            android:layout_width="93dp"
android:layout_height="49dp"
```

```
            android:id="@+id/forward"
            android:text="@string/Forward"

android:textColor="@color/common_google_signin_btn_text_dark_default"/>

<RadioButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/reverse"
            android:text="@string/Reverse"

android:textColor="@color/common_google_signin_btn_text_dark_default"/>

</RadioGroup>

<ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:srcCompat="@drawable/irn_bru"
        android:id="@+id/imageView"
        android:layout_above="@+id/skb"
        android:layout_alignLeft="@+id/myRadioGroup"
        android:layout_alignStart="@+id/myRadioGroup"
        android:layout_alignRight="@+id/myRadioGroup"
        android:layout_alignEnd="@+id/myRadioGroup"/>

<Switch
        android:text="ON/OFF"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/Power"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"/>

</RelativeLayout>
```

Main Activity

```
package esteban.miguel.irnbrumotor;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.RadioGroup.OnCheckedChangeListener;
import android.widget.SeekBar;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.Toast;
```

```
import java.io.IOException;
import java.io.OutputStream;
import java.util.Set;
import java.util.UUID;

publicclass MainActivity extends Activity {
//DEFINE VARIABLES
private RadioGroup radioGroup;
private RadioButton forward, reverse;
private TextView textView;
private OutputStream outputStream;
private SeekBar miSeek;
private TextView FrequencyText;
private Switch Power;

    @Override
protectedvoid onCreate(Bundle savedInstanceState){
super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


            //MATCH VARIABLES WITH ID OF THE BUTTONS DEFINED IN LAYOUT
        forward =(RadioButton) findViewById(R.id.forward);
reverse =(RadioButton) findViewById(R.id.reverse);
        textView =(TextView) findViewById(R.id.text);
        miSeek =(SeekBar) findViewById(R.id.skb);
FrequencyText =(TextView) findViewById(R.id.FrequencyText);
        Power =(Switch) findViewById(R.id.Power);
        radioGroup =(RadioGroup) findViewById(R.id.myRadioGroup);


//BLUETOOTH CONNEXION
        BluetoothAdapter bluetoothAdapter =
BluetoothAdapter.getDefaultAdapter();
if(bluetoothAdapter ==null){
            Toast.makeText(getApplicationContext(),"Device doesnt Support
Bluetooth", Toast.LENGTH_SHORT).show();
}

if(!bluetoothAdapter.isEnabled())

{
            Intent enableAdapter =new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableAdapter,0);
}

        Set<BluetoothDevice> bondedDevices =
bluetoothAdapter.getBondedDevices();
        BluetoothDevice device =null;

if(bondedDevices.isEmpty()){

            textView.setText("Please Pair the Device first");

}else{

for(BluetoothDevice iterator : bondedDevices){
```

```
if(iterator.getName().equals("HC-05"))
            //Replace with iterator.getName() if comparing Device names.
{

                    device = iterator;

boolean found =true;

break;

}
}
}
        UUID myUUID = UUID.fromString("00001101-0000-1000-8000-
00805F9B34FB");//BT  module
try{
            BluetoothSocket socket
=device.createRfcommSocketToServiceRecord(myUUID);
            socket.connect();
outputStream = socket.getOutputStream();
}catch(IOException e){
            e.printStackTrace();
}
//END OF BLUETOOTH CONNEXION

// START/STOP SWITCH
        Power.setChecked(false);
        Power.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener(){
            @Override
publicvoid onCheckedChanged(CompoundButton buttonView,boolean isChecked){
if(isChecked){
try{

                        outputStream.write(1);
 }catch(IOException e){
                        e.printStackTrace();
 }
                    textView.setText("Start");
 }else{
 try{
                        outputStream.write(0);
 }catch(IOException e){
                        e.printStackTrace();
 }
                    textView.setText("Stop");
 }

 }

 });

             //SEEK BAR FREQUENCY SELECTION
             //Initial frequency value display
         FrequencyText.setText("Frequency: "+(miSeek.getProgress()+20)+"
 Hz");
             //Seek bar from 0 to 40--> Add 20 to transform into frequency
             //Three interrupts.
             //On start --> Do nothing
             //On progress --> Display seek bar value
             //On stop --> Send the byte value via Bluetooth
```

```
        miSeek.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener(){
            @Override
publicvoid onProgressChanged(SeekBar seekBar,int progress,boolean
fromUser){
               FrequencyText.setText("Frequency: "+ String.valueOf(progress
+20)+" Hz");
}

            @Override
publicvoid onStartTrackingTouch(SeekBar seekBar){

}

            @Override
publicvoid onStopTrackingTouch(SeekBar seekBar){
try{
int FreqValue = miSeek.getProgress()+20;
                    outputStream.write(FreqValue);
}catch(IOException e){
                    e.printStackTrace();
}
                FrequencyText.setText("Frequency:
"+String.valueOf(miSeek.getProgress()+20)+" Hz");
}
});

//RADIO GROUP FOR DIRECTION OF THE MOTOR
        radioGroup.setOnCheckedChangeListener(new
OnCheckedChangeListener(){
            @Override
publicvoid onCheckedChanged(RadioGroup group,int checkedId){
// Find out which radio button is selected comparing their ID
if(checkedId == R.id.forward){
try{
                        outputStream.write(2);//Forward
}catch(IOException e){
                        e.printStackTrace();
}
                    Toast.makeText(getApplicationContext(),"Changing to
rotate forward",Toast.LENGTH_SHORT).show();
}else{
try{
                        outputStream.write(3);//Reverse
}catch(IOException e){
                        e.printStackTrace();
}
                    Toast.makeText(getApplicationContext(),"Changing to
rotate reverse",Toast.LENGTH_SHORT).show();
}
}

});
}
}
```