Hochschule Niederrhein
University of Applied Sciences

Elektrotechnik und Informatik
Faculty of Electrical Engineering and Computer Science

DESIGN AND SIMULATION OF A SINGLE-PHASE INVERTER WITH DIGITAL PWM

Bachelor Final Thesis

Industrial Technologies Engineering

Author: Elisa Braco Sola

Supervisor: Pr. Andreas Waldhorst

Krefeld, Germany. July 2016.

Abstract

The current project has as major aim the design of a single-phase inverter for educational purposes. The main distinctive feature is the digital implementation of the PWM modulation.

Since the first Arduino board was developed on 2005, there has been a turning point on the programming world, especially between non specialised users. With an intuitive software and a wide amount of applications, it is a highly recommended option for the first approach to digital programming for students. For those reasons, this paper has the aim to apply this tool to ease the PWM implementation on a single-phase inverter, substituting analogical circuitry.

Although nowadays it is possible to find integrated circuits with single – phase inverters, in order to reinforce the educational approach, the circuit will be designed and built piece by piece. Thus, it is easier to understand and visualize all the components required, as well as to check the waveforms obtained on the internal elements.

To achieve those aims, a first complete theoretical analysis will be made, including its applications and basic elements. Afterwards, the specific characteristics of the desired inverter will be defined, allowing the computation and selection of the components required.

A fundamental part of the work relies on the programming of the PWM signal. For this reason, an insight into the possibilities provided by the microcontroller will be made. Some options for codes will be developed and the optimal for the application will be selected.

After the theoretical approach, the complete circuit will be simulated with the LTSpice software and implemented in a protoboard. Some measurements will be also done in order to check the performance of the device and its efficiency. Finally, a PCB with the complete circuit will be developed.

Key words: single-phase inverter, PWM, Arduino.

Resumen

El presente proyecto tiene como objetivo principal el diseño de un inversor monofásico con fines académicos. El rasgo más distintivo es la implementación digital de la modulación PWM.

Desde que el primer Arduino fuera desarrollado en 2005, el mundo de la programación ha sufrido un punto de inflexión, especialmente entre usuarios no especializados. Con un software intuitivo y un amplio abanico de aplicaciones, es una opción muy recomendable para el primer contacto de los estudiantes con la programación digital. Por ello, este documento pretende aplicar esta herramienta para facilitar la implementación PWM en un inversor monofásico, sustituyendo la tradicional circuitería analógica.

A pesar de que hoy en día es posible encontrar inversores monofásicos en circuitos integrados, con el fin de reforzar el enfoque educacional, el circuito será diseñado y construido pieza a pieza. Así, se facilita el entendimiento y visualización los componentes necesarios, así como la comprobación de las formas de ondas en los elementos internos.

Para lograr estos objetivos, se realizará un primer análisis teórico, incluyendo aplicaciones y elementos básicos. Posteriormente se definirán las características específicas del inversor deseado, permitiendo así el cálculo y selección de los componentes necesarios.

Una parte fundamental del trabajo recae en la programación de la señal PWM. Por ello, se contemplarán las posibilidades del microcontrolador al respecto. Diversas opciones para el código serán consideradas y se seleccionará la mejor para la aplicación.

Tras el enfoque teórico, el circuito completo será simulado con el software LTSpice e implementado en una protoboard. Asimismo se realizarán medidas con el fin de comprobar el correcto funcionamiento del equipo y su eficiencia. Finalmente, el circuito será impreso en una PCB.

Palabras clave: inversor monofásico, PWM, Arduino.

Acknowledgments

I would like to thank my supervisor Prof. Dr.-Ing. Andreas Waldhorst for the tracking and support throughout the project, as well as Alexander Lennartz and Lasse Wagner for all their advice and patience.

A special thanks goes also to the University of Applied Sciences – Hochschule Niederrhein for providing me the material required and allowing me to use its facilities, and to the Universidad Pública de Navarra for these four years of knowledge and training as engineer.

Elisa Braco

Krefeld, Germany. July, 2016.

Contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Objectives

The main objective of this project is the design, simulation and testing of a single-phase inverter for educational purposes. In order to achieve this, the first step is the **analysis** of the device, with a full regard into topologies, modulation and components.

After the definition of the performance criteria, the **real elements** have to be selected. Therefore some computations will be necessary to determine the concrete parts.

The aim is to implement the **PWM modulation** digitally, and for this purpose, the possibilities provided by the microcontroller will be analysed.

Once the **theoretical simulation** is made, the circuit should be built into a **protoboard** for real testing. As a final step, a **PCB model** of the circuit will be developed.

1.2 Scope

This project follows all the steps necessary to design a real device. First of all a **theoretical approach** into inverters and their parts is made in Chapter 2 in order to have a first idea of the circuit.

Chapter 3 goes into the **design** with detail, with a complete discussion and selection of all the elements needed. Also, the requirements for the inverter are set, in order to measure its quality.

One of the key aspects is the **microcontroller**. Therefore, Chapter 4 focuses on its characteristics and explains the possibilities available regarding PWM generation.

After the theoretical development, on Chapter 5 the **results** obtained on the real circuit are collected. Also a computer software is employed to simulate them in order to have a better idea of the results expected.

Finally, Chapter 6 collects a **resume** of the objectives achieved, as well as some recommendations and guidelines for further improvements on the project.

The final **appendixes** gather a complete explanation of the calculations made, the complete code implemented and the technical documents of the work.

Chapter 2

Single – Phase Inverter

This chapter presents the general topic of this work: the **single phase inverter**. Not only definition, but applications and components are described in those pages.

The main **topologies** available for inverters will also be discussed, in order to justify the final decision of a Full Bridge configuration. Also, an insight into its main component, the **elementary switching cell**, is given.

Within the elementary switching cell, **semiconductors** are the devices responsible for its operation. Due to their importance, a brief description of the main types, covering structures and schematics is included. Also, a final comparison depending on frequency and power can be found.

An important block in this chapter is the **PWM modulation**. After a first definition and explanation, the two main options for modulation are described: bipolar and unipolar.

Finally, other parts of the inverter are commented, as the **driver** and the **filter**.

2.1 Definition

A power inverter, or inverter, is an electronic device or circuitry that changes **direct current** (DC) into **alternating current** (AC).

Depending on the number of phases of the AC output, there are several types of device. **Single-phase** and **three-phase** inverters are the most common configurations. Figure 2.1 shows a schematic of their basic performance.

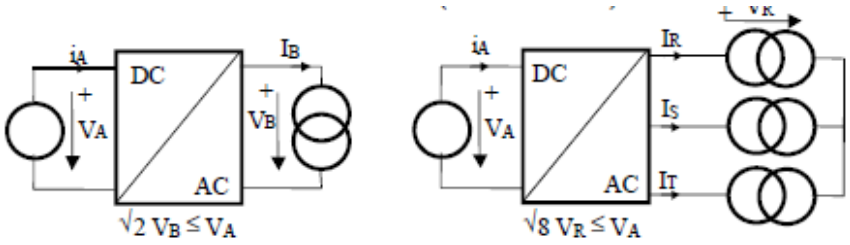


Figure 2.1: Single - phase and three – phase voltage inverters [1]

2.2 DC - AC current

Technically, there are two ways in which electricity can be transmitted: direct current and alternating current.

Direct Current (DC) is the unidirectional flow of electric charge. If a constant voltage is applied across a circuit, it results in a constant current. DC can be produced by different sources, being batteries the most common ones. Also dynamos, power supplies or solar cells are examples of DC generators.

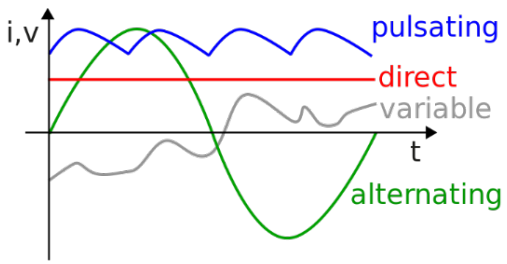


Figure 2.2: Direct and alternating current and voltage [2]

On the other hand, in **Alternating Current (AC)** the flow of electric charge periodically reverses its direction. The most usual waveform of AC is a sine wave, but it can also be triangular or square wave.

In order to transfer electrical power into different kinds of current, special devices are required. Converters AC/DC are called **rectifiers** and **inverters** are DC/AC converters. It is important to do this transformation in the most efficient way in order not to lose energy.

Another important issue to consider is **frequency**. In a DC voltage, given that the voltage value is constant, frequency is zero. However, when AC voltage is involved, it is necessary to define the number of repetitions per unit of time of a periodic event, that is to say, the frequency. **Period** is the inverse of frequency, and it can be defined as the amount of time required in order to complete a repetitive cycle.

2.3 Applications

When it comes to transform a source of Direct Current into an Alternating Current, the amplitude of output and input do not need to be the same. The most widespread converters are the **voltage inverters**, also called power-reducers, as the output voltage is lower than the input.

Sometimes combined with boost converters, inverters are used in a wide range of applications, as injection of energy to the grid on renewable energies, isolated generation or electric drives.

An example for this is electricity generation with solar cells. When sun light reaches a cell, electrons are released. Applying an electric field, they can be taken out and result in a DC current. In order to put this electricity on the grid, an inverter is needed. Another example is an AC device connected to a battery as power source. Given that direct connection is not possible, an inverter is required as an intermediate device.

Figure 2.3: Examples of applications [3]

2.4 Output: modified sine wave and pure sine

With regard to their output, two different types of power inverters can be found: modified sine wave and pure sine wave.

On **modifying sine wave** inverters, the output waveform is similar to a square but with a null period between positive and negative values. Due to its high distortion levels, it is not commonly used, as some devices do not work properly with it.

On the other hand, a **pure sine inverter** provides a sinusoidal waveform, requiring more complicated circuitry but assuring better efficiency and lower distortion than the previous type. The majority of the inverters are pure sine inverters, given that their output can be applied to all loads.

The differences between both waveforms can be appreciated in Figure 2.4. Also, a square wave is included in order to establish a reference for comparison. It can be seen that the modified sine is the intermediate waveform between the square and the sine. While modified sine rises or falls instantaneously after sitting at zero for a moment, the pure sine waveform passes through all values between minimum and maximum, resulting in a smoother input for the load.

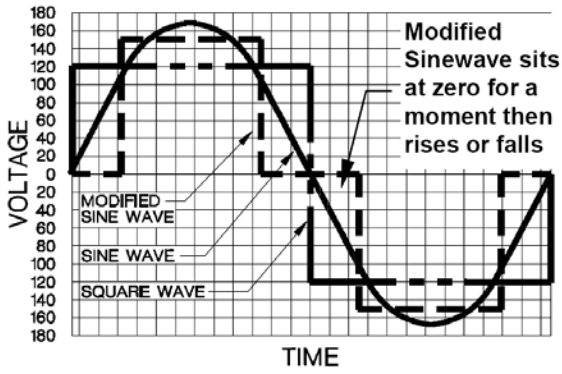


Figure 2.4: Square, modified and sine wave [4]

2.5 Elementary switching cell

The main component of an inverter is the **elementary switching cell**. It connects an input and an output source and basically consists of two switchers. Regarding the sources, some basic electricity rules must be followed:

- Two current sources cannot be connected in series unless they have the same value. Otherwise charges start to be accumulated, as the principle of charge conservation is not achieved.

- Two voltage sources of different value cannot be connected in parallel. In case this happens, there is a voltage drop and a current between the sources appears. The final voltage would be equal for both sources but in no case the highest value of voltage would be achieved.

Consequently, one source must be of **current** and the other of **voltage**. Also it is necessary to take into account that a voltage source cannot be short-circuited (with an inexistent load the current would increase only limited by parasitic resistances) and that a current source cannot be left in open circuit (the charges would not flow). Thus, there are only **three possible connections** between the sources, as shown in Figure 2.5. Energy flows only on states 1 and 3.

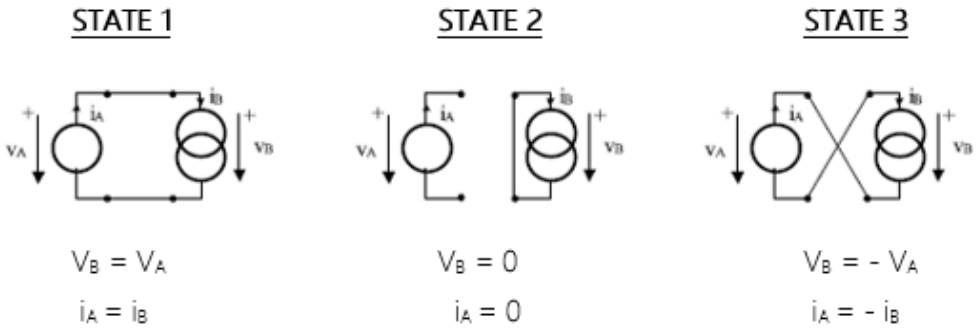
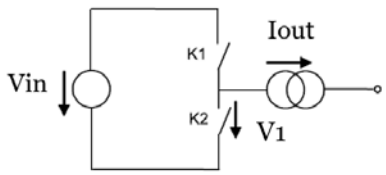


Figure 2.5: Connection states allowed [1]

In case the current source is the input, the converter is a **step-up (boost)**. Otherwise, if the voltage source is the input, the converter works as a **step-down (buck)**.



In an elementary switching cell, the interrupters cannot be opened or closed at the same time, so they must be **complementary in performance**. Therefore, only one connection function is required for both.

Figure 2.6: Elementary switching cell

Table 2.1 is an example of the output voltage regarding different values of the connection function. The switch K_1 is connected to F and the input voltage is Vin.

Connection function F	K1	K2	V1
1	1	0	Vin
0	0	1	0

Table 2.1: Resume of states in an elementary switching cell.

2.6 Semiconductors. Selection of interrupters

Power semiconductors are the main part of the switching cell. The inverter's features depend largely on their performance and characteristics.

Basically, all semiconductors consist of **PN junctions**. Those are achieved doping silicon with boron (P layer) or phosphorus (N layer). When a certain voltage is applied to the junction, the internal charges move, letting the current flow through it. If the voltage is not enough, the charges do not flow and therefore the semiconductor acts like a blocker. An ideal semiconductor would perform as an **ideal switch**, being its main characteristics on- and off-control, instantaneous switching, null impedance in conducting and infinite impedance in cut-off. Also, reversibility regarding current and voltage and the voltage range in conducting are issues that define a semiconductor.

In this section, a brief insight into different semiconductors will be given.

2.6.1 Types of semiconductors

- **DIODE**

A diode consist of a single PN junction. It is the simplest semiconductor and has no on- and off-control. Its symbol and structure are reflected in Figure 2.7.

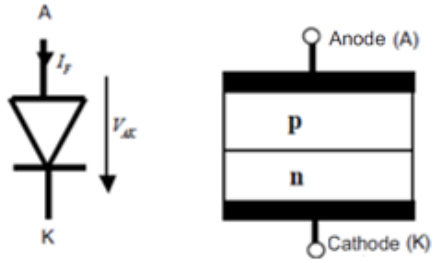


Figure 2.7: Symbol and structure of a diode [1]

When a positive voltage over a specific value (Threshold voltage) is applied between P (anode) and N (cathode) it conducts, whereas a negative voltage results in the cut-off of the diode. However, there is a limit for negative voltage applied, as the PN junction can be broken. Figure 2.8 shows the **static characteristic** of a diode. Three main areas can be observed: forward direction (diode conducting), blocking area (diode cut) and break through (diode destroyed).

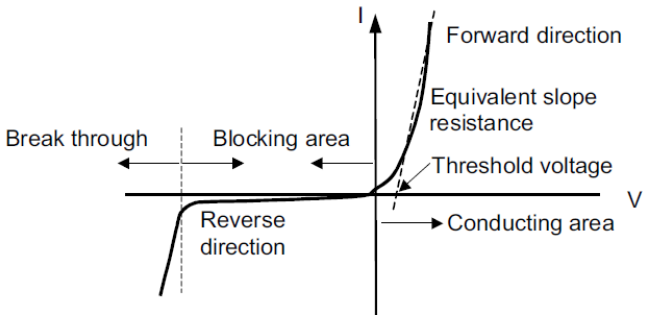


Figure 2.8: Current-voltage characteristics for a diode [5]

- **BJT**

Two PN junctions result in a **Bipolar Junction Transistor** (BJT). The basis of this semiconductor is to amplify current. Applying a positive voltage between collector (C) and emitter (E) is not enough to make it conduct. Therefore, it is required to introduce a positive current on the base in order to achieve current circulation, with a base-to emitter voltage (V_{BE}) over the threshold value.

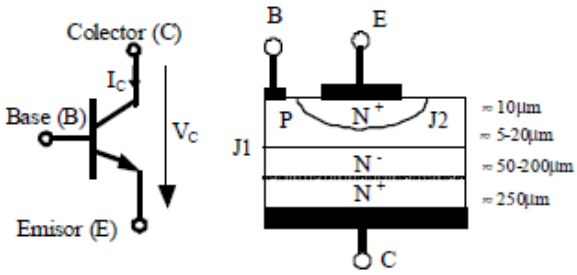


Figure 2.9: BJT symbol and structure [1]

BJTs have current gains in the order of 10 for comparatively high voltage drops. As one of their drawbacks, it is necessary to introduce high currents on the base in order to make the device work in the forward-active area (amplification of current). Also, high reverse base drive currents are required to obtain a fast turn-off. For these reasons, they are not commonly used in power electronics.

- **MOSFET**

The **Metal Oxide Semiconductor Field Effect Transistor** or MOSFET is a voltage controlled semiconductor, unlike BJT, that is current controlled.

There are two types of MOSFETs: channel N and channel P, but this last one is not commonly used in power electronics. The voltage control is between gate and source, and normally must be above 10 Volts in order to switch on the semiconductor. Structure and symbol of an N-Channel MOSFET are shown in Figure 2.10.

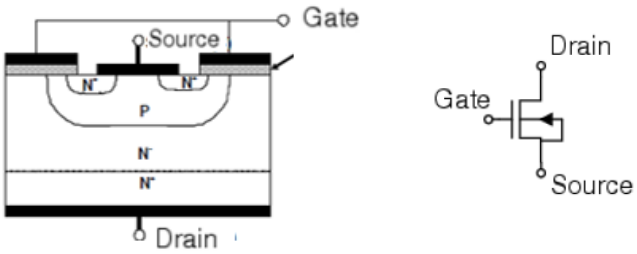
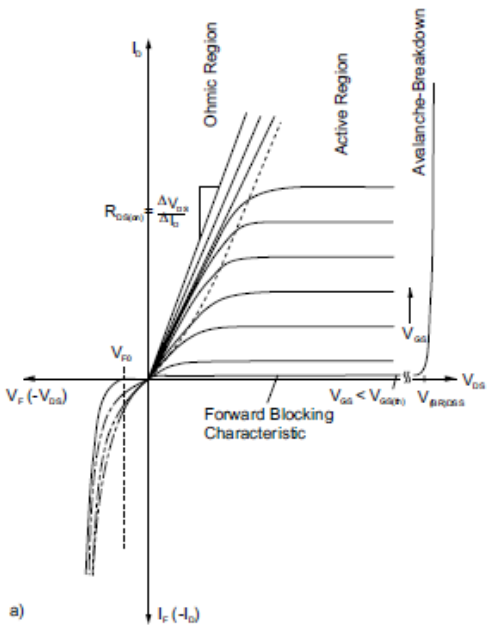


Figure 2.10: Structure and symbol of an N-Channel MOSFET [1]

Basically, in order to turn it on, a voltage between gate and source (V_{GS}) must be applied. This fact, together with a positive voltage between drain and source (V_{DS}), result in an electron channel that allows a current flow in the drain (I_D).

This semiconductor is commonly used in low voltage and low power applications, as microelectronics. Also, when a high switching frequency is required.



Regarding their static behaviour, reflected in figure 2.11, MOSFETs can operate between the breakdown voltage and the maximum direct voltage. Below and over these values, the semiconductor gets into avalanche and the junctions are destroyed. When conducting in the ohmic region, MOSFETs behave as a resistor, amplifying the current. On the other hand, in the active region the current is linear.

Figure 2.11: Static behaviour of an N-Channel MOSFET [5]

- **POWER-MOSFET**

As a concrete type of MOSFETs, Power MOSFETs are specially designed to handle higher power levels. Due to their low gate drive power, fast switching speed and good paralleling capability, they are commonly used in power electronics.

The most common structure in Power MOSFETs is the **Vertical Diffused MOS (VDMOS)**, with the source electrode over the drain, which results in a vertical current when the device is conducting. Whereas Lateral Diffused MOS (LDMOS) are mainly used in high-end audio amplifiers, VDMOS are the option normally selected for switching applications.

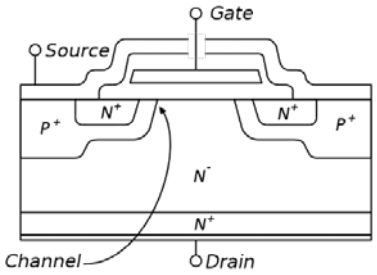


Figure 2.12: Cross section of a VDMOS, showing an elementary cell [27]

The internal structure of an elementary cell of a VDMOS is shown in Figure 2.12. The complete semiconductor is composed of several thousands of cells.

The source metallization connects N^+ and P implantations, creating a diode between the drain (cathode) and the source (anode) of the MOSFET. This **body diode** can be employed as freewheeling diode in H-Bridge configurations.

The main difference between Power MOSFETs and normal MOSFETs is the current capacity and the gate capacitance. Whereas normal MOSFETs perform better in higher frequencies, Power MOSFETs are a better option regarding high current and voltage applications. Also, power MOSFETs have much lower resistance while conducting, minimising conducting losses.

- **IGBT**

The **Insulated Gate Bipolar Transistor** is similar to a MOSFET but with a third PN-junction. This allows controlling it with voltage, as a MOSFET, but with output characteristics similar to a BJT regarding high loads and low saturation voltage. Its schematic structure and symbol are reflected in Figure 2.13.

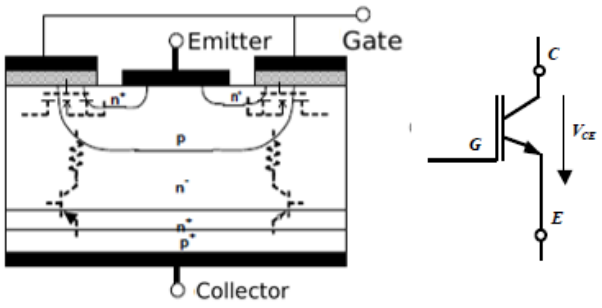


Figure 2.13: Structure and symbol of an IGBT [1]

Four main regions can be observed on its **static behaviour**. From the left to the right in figure 2.14, the avalanche region is the area when a voltage below breakdown voltage is applied, resulting in the destruction of the IGBT. The **cut area** includes values from breakdown voltage up to threshold voltage. The IGBT does not conduct in this region. On the **saturation** region, the IGBT behaves as a voltage source and a series resistance. With low variations of voltage, high amplification of current can be achieved. This area is the most desirable for working. If the voltage is augmented, the IGBT enters in **active** region, and current remains constant. There is a maximum voltage applied in order the IGBT not to enter in avalanche.

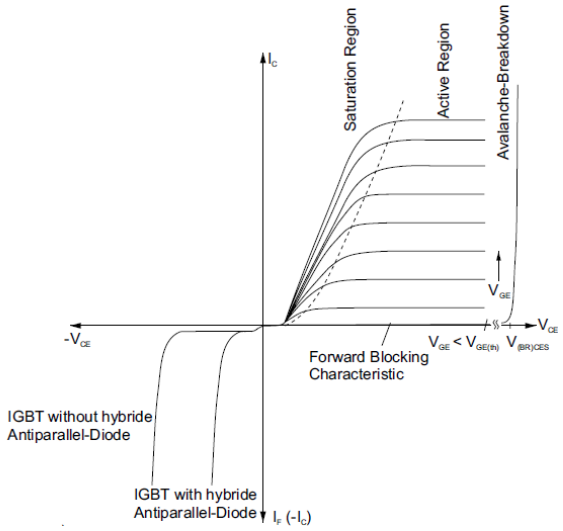


Figure 2.14: Static behaviour of an IGBT [7]

It is the most used semiconductor in power electronics, given that it can support a wide range of voltage from few volts to kV and powers between kW and MW. Also, in applications with switching frequencies lower than 20 kHz, it is the semiconductor commonly selected.

- **Other semiconductors**

IGBTs and MOSFETs are the most used semiconductors in power electronics. However, especially in very high power, other semiconductors like THYRISTORs or GTOs are employed.

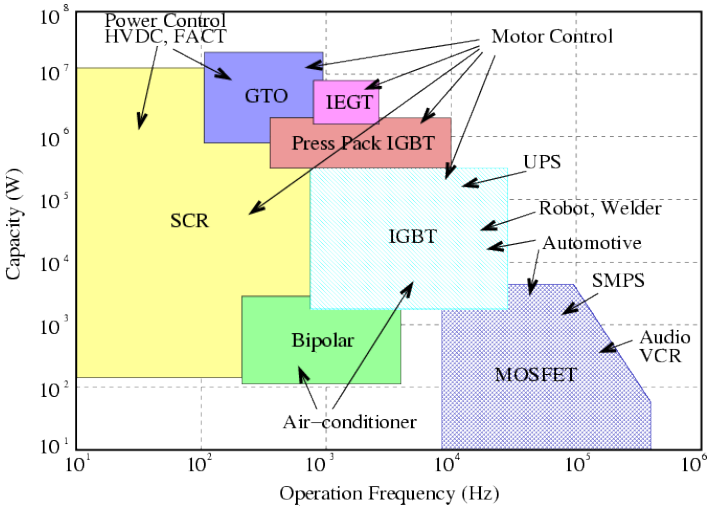
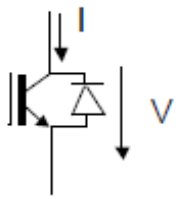


Figure 2.15: Selection of semiconductors regarding capacity and operation frequency [6]

In Figure 2.15, a selection of semiconductors depending on power and operation frequency is resumed. Regarding power, at low frequencies SCR (a type of THYRISTORs) are the most used. As frequency increases, they are replaced by GTOs or IGBTs. Finally, with high frequencies MOSFETs are the semiconductors selected, as their switching losses are lower comparing with the rest. From the graph, it is remarkable the lack of semiconductors with both high frequency and high power performance. The difficulty of combining both features is directly related to losses and heat evacuation.

2.6.2 Selection of semiconductors for an inverter

The selection of the interrupters depends on the reversibility required for the voltage and the current source. On an inverter, it is only necessary current reversibility. Hence, an interrupter of **three segments** must be chosen.



Selecting a **transistor and a diode in antiparallel**, the turning-on of the transistor can be controlled with voltage. When the voltage applied is positive and an on-order is given, the transistor conducts and the current is positive. If the voltage is negative, the diode conducts and the current is negative.

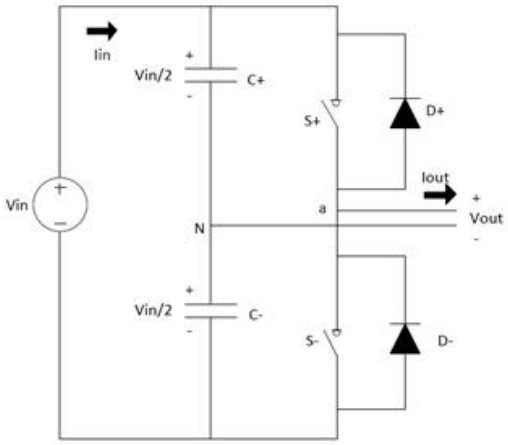
Figure 2.16: Semiconductors chosen

Whereas the diode is always in a switching cell, different **transistors** can be chosen. Criteria applied for the decision regards applications, switching frequency, velocity and voltage required. On Chapter 3, a discussion about the best semiconductor to choose will be held.

2.7 Inverter topologies: Half - Bridge and Full - Bridge

Regarding inverters, there are several topologies available, being the most common ones Half Bridge and Full Bridge. This section collects a brief explanation of them.

2.7.1 Half - Bridge



In this topology, only **two switchers are required**. The DC input is divided in two identical sources and the output is referenced to the middle point.

Figure 2.17: Single-phase Half-Bridge voltage inverter. [7]

A **capacitor divisor** is used to achieve the medium voltage point (N). By controlling the voltage in N, direct current injected in the alternate side is assured to be zero.

In addition to this, only one switching cell is required. Therefore, it is a cheap alternative and the conducting losses are not excessively high.

On the other hand, in order to obtain the same value of power, higher currents are required, as voltage is lower. In case a high voltage is needed in the output, an elevator is commonly used as first step, as the input voltage must be double than the output desired. Regarding switching losses, the semiconductors must be designed for 2Vout. This fact makes this topology the **worst in performance**, as switching losses become excessively high.

2.7.2 Full H-Bridge

In a Full H-Bridge, the alternate output voltage (Vab in Figure 2.18) is obtained by the difference between two branches of switching cells. Therefore, **four switchers** are needed. To maximize the fundamental component of the output voltage, the fundamental component of the voltage on each branch (Vao and Vbo) must be 180º out of phase. The semiconductors of each branch are complementary in performance, which is to say when one is conducting the other is cut-off and vice versa. This topology is the most widely used for inverters.

The semiconductors must be designed only for Vout, but as a disadvantage, four switchers are required and therefore, losses can become elevate.

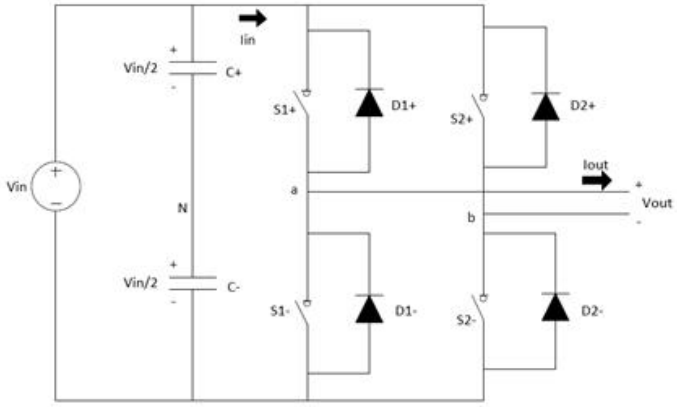


Figure 2.18: Single-phase Full H-Bridge voltage inverter [7]

2.8 PWM Modulation

In order to obtain the connection function for the semiconductors, a modulation technique called PWM is employed. The basics of **Pulse Width Modulation** (PWM) is the variation of the duty cycle of a periodic signal. The duty cycle (D) is defined as the variation of the positive part of the signal (T_{on}) related to the period (T).

$$D = \frac{T_{on}}{T}$$

In analog circuitry, a PWM is obtained comparing a signal control (V_{con}) with a triangular wave. For this purpose, normally a comparator is used. Thus, the output will be 1 if the positive leg of the comparator is higher than the negative one and 0 if it is lower. Figure 2.19 reflects schematic and waveforms obtained in an analogical PWM circuit.

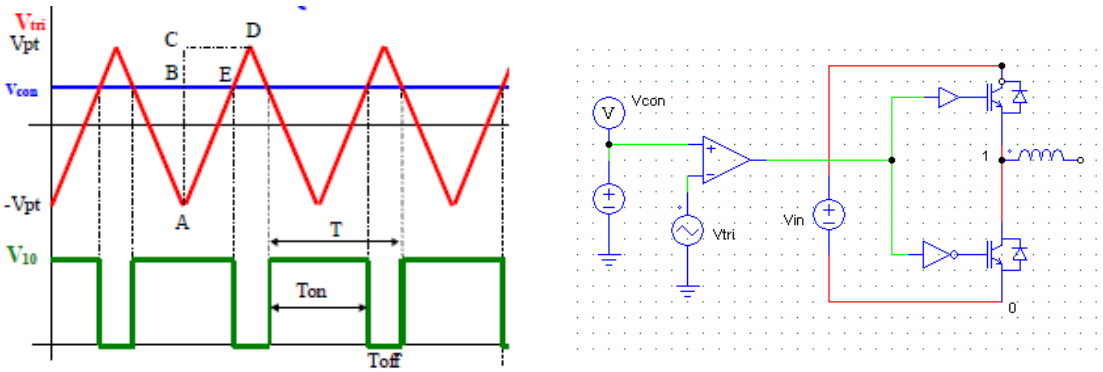


Figure 2.19: PWM waveforms and schematic.

It must be pointed out that with constant Vcon only a fixed duty cycle can be obtained. For the purpose of having a **varying duty cycle**, the signal control must be **sinusoidal**. This way, as Vcon is not constant, the positive part relating to the period changes. Hence, two frequencies must be considered. Vcon oscillates normally at the **output desired frequency**, which is usually a low value, 50 or 60 Hz. Also, it is necessary to determine the frequency of the triangular wave. This frequency is directly related to the **switching frequency** of the semiconductors, and must be carefully chosen. Normally, the value selected is over 20 kHz, in order to overcome the human hearing threshold.

Basically, there are two main techniques to obtain a PWM: bipolar and unipolar modulation. Both can be applied to a Full H-Bridge, and will be studied in the following section. The high-side interrupters are named K1 and K3 and the low-side are K2 and K4.

2.8.1 Bipolar Modulation.

In bipolar modulation, the output voltage oscillates between two values (Vdc and –Vdc in Figure 2.20), hence its name.

A basic schematic of a Full H-Bridge with bipolar modulation is collected in Figure 2.20. It can be seen how in this modulation, the connection function is the same for K1 and K4 and for K2 and K3. This means, only **one signal control is required**. As the switchers are complementary in performance on each branch, the connection function must be inverted in order to achieve this.

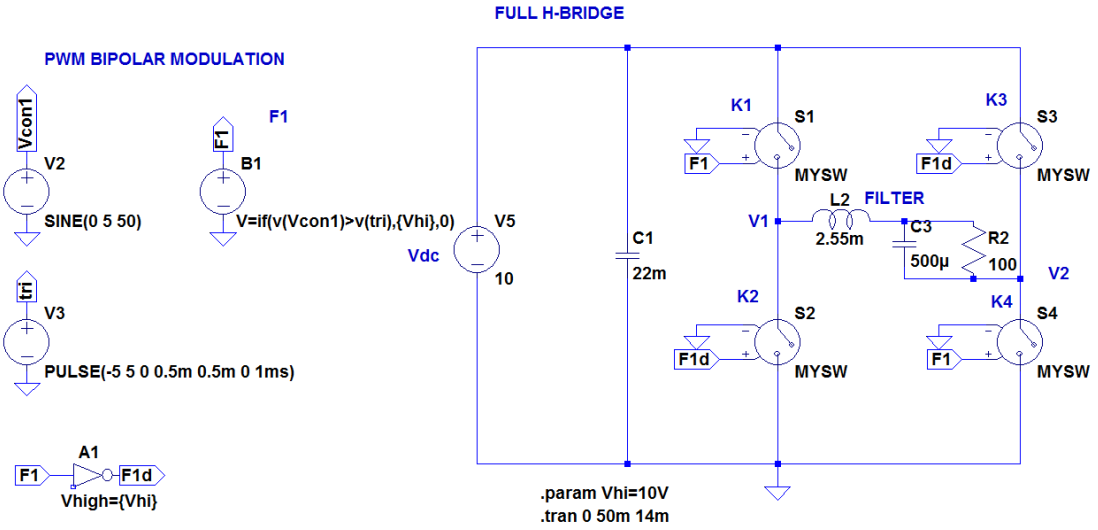


Figure 2.20: Analogic circuitry for bipolar modulation in LTSpice software

The connection function, and therefore the switching of the MOSFETs depend on the difference between the control voltage (Vcon) and the triangular wave (Vtr). The following schematic resumes the performance of bipolar modulation.

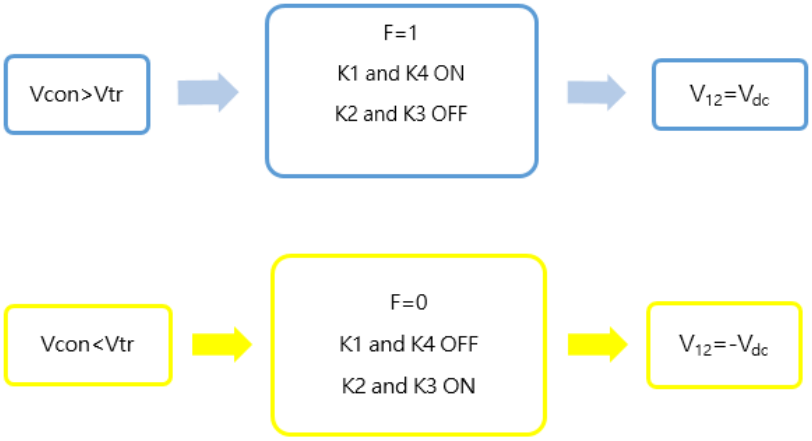


Figure 2.21: Resume of bipolar modulation.

Also, the main waveforms are collected in Figure 2.22: Vcon (sinusoidal control voltage), Vpt (triangular voltage) V10 and V20 (voltage on the low side switchers of each branch), V12 (output voltage without filtering) and iL (current on the inductance).

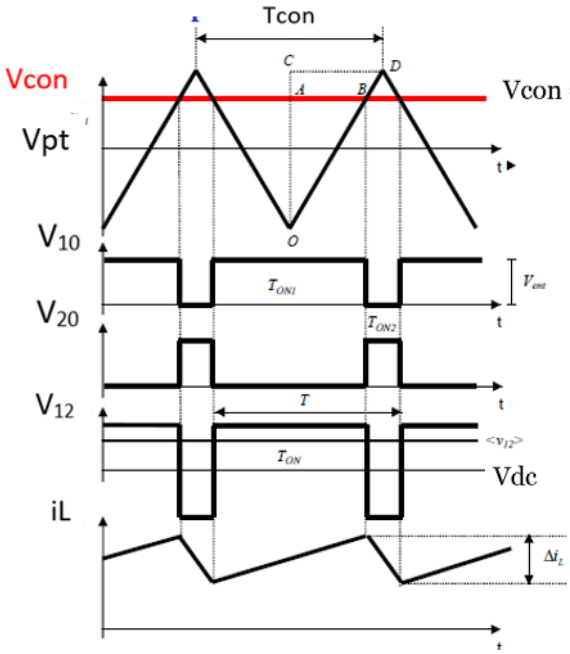


Figure 2.22: Main waveforms of bipolar modulation. [1]

The output voltage (V12) results from the difference between voltages in both legs. It oscillates between **two levels**, Vdc and –Vdc. Its period is the same as the switching one, as it can be seen in Figure 2.22.

In bipolar modulation, the output voltage **switching harmonics** are the sum of the switching harmonics of both branches. In order to reduce them, an option is increasing switching frequency, as this eases the filtering. However, as a result the switching losses may become too large.

An important advantage of this modulation is the **constant voltage to ground** (common mode voltage), which make bipolar modulation the one preferred in applications as photovoltaic systems without transformer.

2.8.2 Unipolar Modulation.

Unlike bipolar modulation, in case unipolar modulation is implemented in a circuit, the connection function is **different in both branches**. Given that the switchers must be complementary in performance, both branches of the bridge are related by the triangular voltage. Furthermore, the sinusoidal control voltage is opposite in sign in both legs.

$$V_{con1} = -V_{con2}$$

Figure 2.23 shows a simple schematic of the implementation of unipolar modulation with LTSpice software.

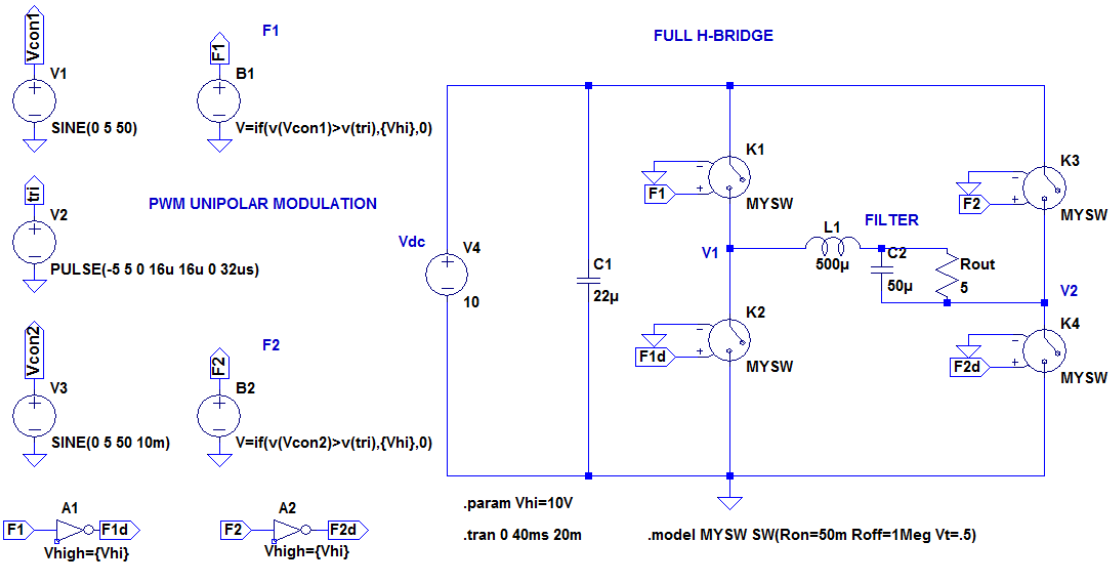
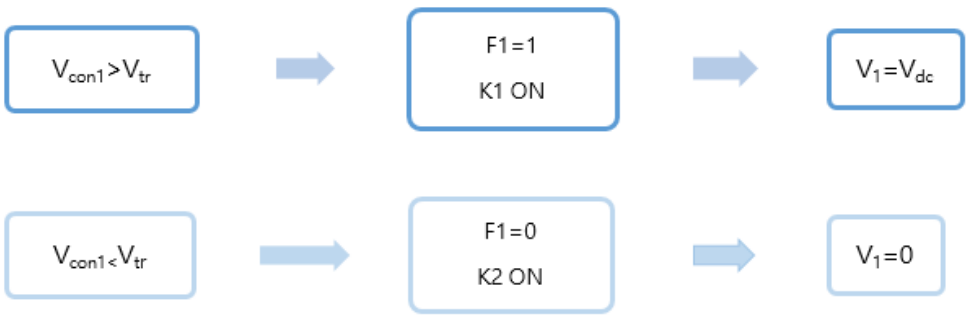


Figure 2.23: Analogic circuitry for unipolar modulation in LTSpice software.

The operation of the connection functions is the same as in bipolar, but instead of having two levels of voltage in the output, there are **three**: Vdc, 0 and –Vdc. This way, the output is **more accurate** than in bipolar modulation. A resume of the performance of unipolar modulation is collected below (Figure 2.24). Also, the main waveforms can be observed in Figure 2.25.



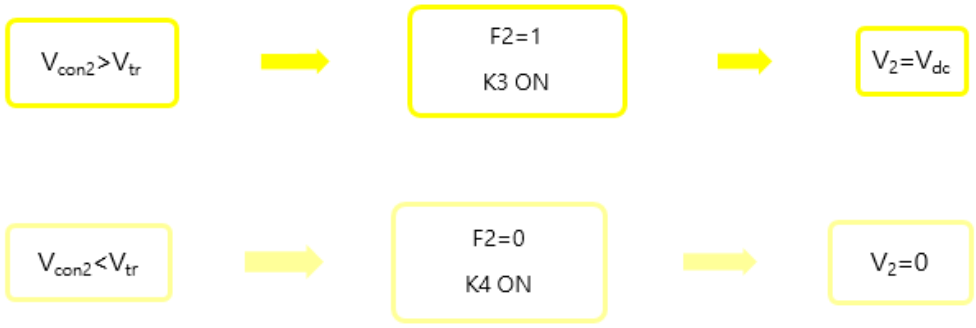
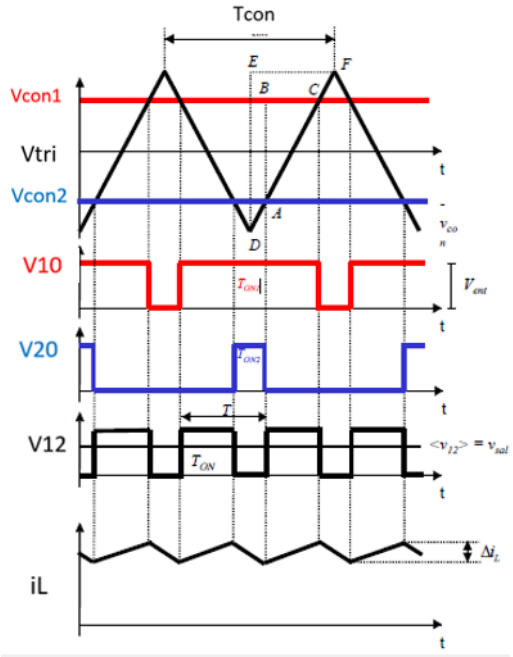


Figure 2.24: Resume of unipolar modulation.

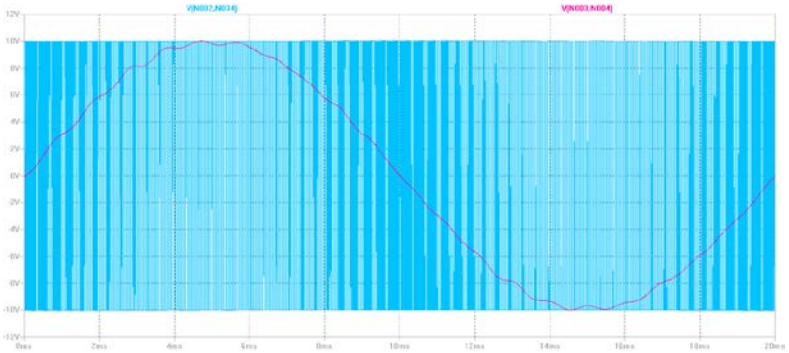


The period of the output voltage is half the switching one and therefore the **output frequency** doubles the switching one. As the **harmonics** are in a higher frequency, for the same switching frequency the output inductance can be smaller than in bipolar modulation.

Also regarding harmonics, as the output voltage results from the difference between branches, the even families of harmonics have double amplitude, given that they are 180º out of phase on each branch. On the other hand, the odd families are in phase and do not appear in the output.

Figure 2.25: Main waveforms of unipolar modulation [1]

Unlike bipolar modulation, unipolar modulation has only one sign on each semiperiod, hence its name. The comparison between both modulations and their outputs is reflected in Figure 2.26. It can be appreciated how the filtered waveform (sinusoidal) is clearer with unipolar modulation. Also the difference between output levels (two in unipolar and three in bipolar) can be seen.



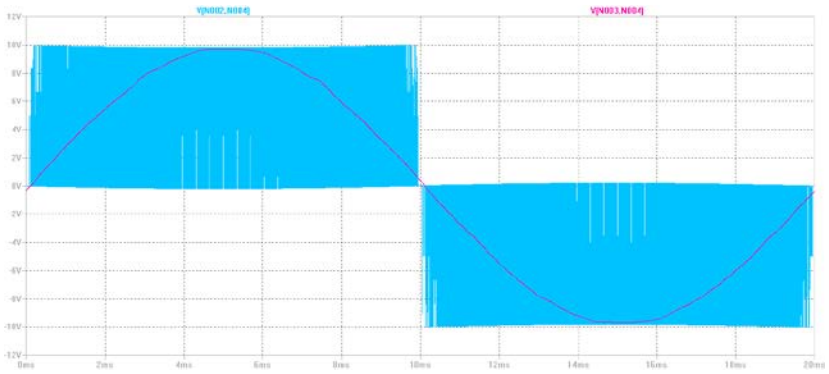


Figure 2.26: Output waveform with bipolar modulation (up) and unipolar modulation.

2.9 Driver

A driver is a device which adapts the connection function to the requirements of the semiconductors. As the connection function only oscillates between two values, sometimes it is necessary to modify the signal in order not to damage the semiconductors.

Main functions:

- Amplification of the control signal in order to adapt it to the desired levels of voltage and current.
- Galvanic isolation: to avoid electrical contact between two parts of the circuit.
- Protection against low feeding voltages or current that could damage the semiconductors.

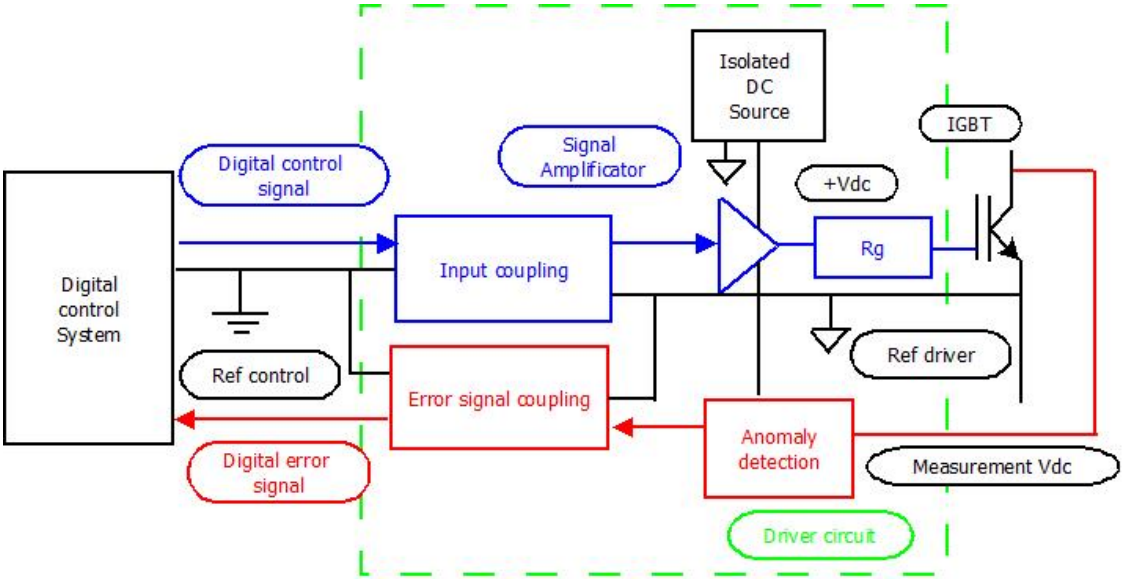


Figure 2.27: Functional schematic of a driver [1]

2.10 Filter

The output voltage is the difference between voltages in the two branches of the bridge. Its waveform is **squared** and therefore a filter is required in order to obtain a sine. Also, it contains a large number of harmonics, which may disturb the correct operation of the output load connected.

A filter is basically a circuit which modifies frequency components between its input and output. It can be analogic or digital and depending on its behaviour, there are mainly four types.

Type of filter	Features
Low pass	They let pass low frequencies and attenuate the ones over a cut point
High pass	They attenuate frequencies below a cut point
Band pass	They let pass frequencies between a range
Band rejection	They block frequencies between a range

Table 2.2: Types of filters and main features.

Once the type of circuit is selected, the cut point and the ranges of frequencies can be chosen by modifying the components.

As in this case the objective is to **attenuate harmonics**, a **low pass filter** will be selected. For a low pass filter, there are two options regarding circuitry: RC (resistor – capacitance) or LC (inductance – capacitance).

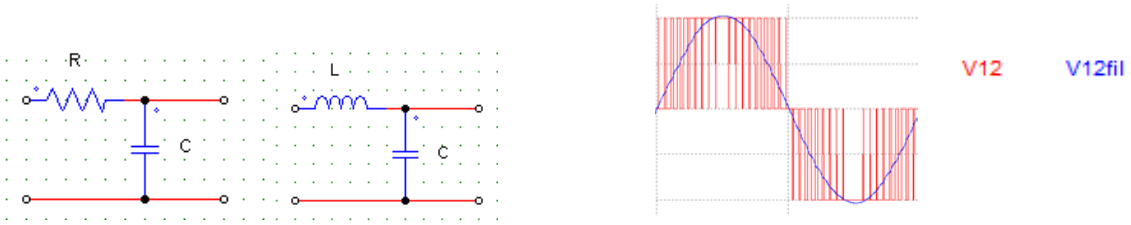


Figure 2.28: RC and LC low pass filters and waveforms.

In Figure 2.28 the output waveform of an inverter before and after an LC filter can be observed. To select the components, it is necessary to determine maximum ripple current, ripple voltage and cut frequency.

2.11 Other projects

Having a look into previous researches on this topic, several options can be found. Between **low voltage single-phase inverter** designs, an important block of projects focus on the generation of analogical PWM. This is the example of "*DC/AC Pure sine wave inverter*" (J. Doucet, D. Eggleston, J. Shaw. Worcester Polytechnic Institute, ref [4]), which provides a detailed description of the analogical PWM generation. On higher voltages, there are various examples of design of inverters, as "*Sistema fotovoltaico aislado: inversor monofásico" (*Barnetxea, A., ref [8]) or "*Diseño e implementación experimental de un inversor monofásico operando en modo isla"* (Martín, R., ref [10]).

All of them have in common that the **PWM generation is analogical** and therefore describe it with detail. On the other hand, this project is focused on the **digital implementation**, also with regard to further control developed on the circuit.

Chapter 3

Design

The aim of this chapter is to gather all the design aspects on an inverter. First of all it is necessary to make an approach to the specific **features** of the problem. Also, a detailed description of the **design criteria** is developed, that is to say, which aspects are the most important regarding the quality of the inverter.

In these pages, a discussion and explanation about different decisions taken in designing is given. After selecting the **topology** of the inverter and its **components**, it comes to compute the different elements of the circuit, as the DC bus capacitor, the output inductance, or the filter. Then, a description of the **losses** in the different semiconductors is made and employing the software LTSpice they are also estimated.

A key aspect in the design is the value of the **switching frequency**. The possibilities available, as well as a discussion about **PWM modulation** are also reflected.

The chapter ends with the **circuit protections**. Not only a description, but also a selection of the components necessary for the adequate operation of the device is made.

3.1 Context (features of the problem)

This project is focused on the design of a voltage inverter for **educational purpose**. Therefore, the power and voltage required are not excessively high. A rough estimate would be between 30 and 50 Watts as maximum power and a voltage range between 5 and 15 volts.

In this context, it can be established a relationship between input and output voltage, as the DC voltage must be $\sqrt{2}$ times the maximum AC voltage. Considering 15 volts as input voltage, the output would be **10.6 effective** volts as maximum.

Regarding frequency, the desired output sine would be of **50 Hz**, standard value in most part of the world.

Also, it is necessary to define a **switching frequency** for the semiconductors. Given that in this work an Arduino will provide the PWM signal, the frequency can be modified as desired. The best results can be achieved on a range from **35 to 62.5 kHz**, fact which will be lately explained.

Therefore, a resume of the **main features** for the inverter is:

- Maximum power: 30 – 50 Watts.
- Input voltage: **15 V (DC).**
- Output voltage: **10.6 V (AC).**
- Sine frequency: **50 Hz.**
- Recommended switching frequency: **35-62.5 kHz.**

3.2 Performance criteria.

A very important aspect in every design is the determination of the performance criteria, that is to say, which aspects are considered the most important regarding the results obtained.

3.1.1 Efficiency

As it was mentioned in the previous chapter, energy conversion should ensure the optimal utilisation. This requirement can be determined by measuring the efficiency of the system. **Efficiency** (η) is defined as the relation between output and input power, usually measured in percentage. Good efficiency ratios are above 80 or 90%.

$$\eta = \frac{P_{out}}{P_{in}} \cdot 100\ (\%)$$

Directly related to efficiency are **power losses**. In semiconductors, losses appear not only while switching but also when conducting due to their internal structure. A deeper insight into losses will be given further in these pages.

A 100% successful conversion is not possible in reality. Although the efficiency rates are usually high, there is always a part of power that dissipates in the form of **heat**. Unluckily, this heat cannot be tapped. Also, it has negative effects on the device, especially regarding performance, as temperature increases and components may not work properly for this reason.

3.1.2 Quality of the waveform

Not only the amount of energy obtained after conversion is important, but also its **quality**. On a pure sine inverter, the waveform should be as clear and pure as possible. There are several factors affecting this waveform, being the performance of the circuitry components an example of that, but mainly the distortion of the sine is due to **harmonics**.

Every periodical signal can be decomposed into a sum of sinusoidal functions, by means of Fourier series expansion. Therefore, the mathematical expression for a periodic voltage is:

$$v(t) = V_{DC} + \sum_{n=1}^{\infty} \sqrt{2}\, V_n \sin(n\omega t + \theta_n)$$

As desired component, the **fundamental component** is the one that oscillates at 50 Hz. It is the first term of the series, and the rest of components are regarded as undesired for several reasons.

V_{DC} is the **DC component** of the voltage. It is the medium value of the signal and normally, this project included, is wished to be zero. In order to achieve this, different techniques can be applied. In the current design, this part of the voltage is removed as a result of the modulation used in the H-Bridge.

Harmonics are components whose frequency is a multiple of 50 Hz. They distort the waveform and their value decreases as the series index increases. Other problems related to harmonics are noise, possible damage in electronic circuits or overcharge of capacitances.

Not all the undesired components have frequencies multiple of 50 Hz. **Inter-harmonics** are named sub-harmonics when they appear below 50 Hz, but can also be found as a broadband spectrum.

The most common way of measuring the quality of waveform is the **Total Harmonic Distortion (THD).** It is defined as the ratio between the sum of power of all harmonic components and the power of the fundamental frequency.

$$THD = \frac{\sqrt{\sum_{n=2}^{40} {V_n}^2}}{V_1}$$

A good voltage THD should be ensured in order to achieve good performance. For low voltage levels, THD should be around 8%, according to UNE-EN 50160 (official standard in Spain that defines features in voltage provided to consumers by electricity distribution grids). However, it is mostly recommended to deal with no more than 5% harmonic voltage distortion factor. The largest single harmonic should not be above **3%** of the fundamental voltage [11]. A high THD can result in malfunction of the equipment.

3.1.3 Reactive Power.

Regarding power, it is necessary to make a first distinction between DC and AC circuits.

On a **DC voltage circuit**, it is only necessary to take into account the power that is transformed into work or heat, called **Active Power** (P). It can be computed as the product of instantaneous voltage (V) and current (I), and its units are Watts. As they are always in phase, power can be calculated as:

$$P = V \,.\, I$$

However, dealing with **AC voltage circuits**, there are more circumstances to be considered. In this kind of circuits, apart from Active Power, there is another one which deals with the formation of electric and magnetic fields in the components of the circuit. This power is not really consumed, but fluctuates between those components and the energy source. Its name is **Reactive Power**, and it is represented with letter Q. Its units are volt-ampere reactive (Var).

The way of relating those two powers is **Apparent Power** (S), which is the vector sum of both Active and Reactive Power. It is measured in Volt-Amperes (VA).

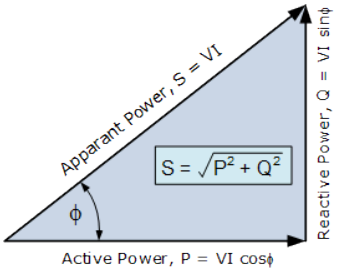


Figure 3.1: Relation between Active, Reactive and Apparent Power [17]

The existence of Reactive Power in an AC circuit depends on the load. If it is purely **resistive**, both current and voltage reverse polarity at the same time, and therefore their product is only positive or zero. They are said to be in phase, and the energy flow goes only in one direction. As a result, only active power is transmitted between the source and the load.

On the other hand, in case the load is **reactive**, voltage and current are out of phase. On each cycle, their product can be positive or negative, and consequently, the energy flow oscillates between the source and the load. This means, an amount of energy is not transmitted to the load (the reactive power). Depending on the gap between current and load, the quantity of energy transmitted varies. This magnitude can be measured by the **power factor (PF)**, which relates active and apparent power.

$$PF = \frac{active\ power\ (P)}{apparent\ power\ (S)} \qquad (1)$$

Equation (1) can also be expressed in terms of the **phase angle** ϑ between powers. As it is reflected in Figure 3.1, active and apparent power can be computed taking into account this angle.

$$P = V\,.\,I\,.\cos\vartheta \qquad\qquad S = V\,.\,I$$

Therefore, equation 1 results in

$$PF = \frac{active\ power\ (P)}{apparent\ power\ (S)} = \frac{V\,.\,I\,.\cos\vartheta}{V\,.\,I} = \cos\vartheta$$

The maximum value for the power factor is 1, that is to say all the energy involved between the source and the load is transmitted. A bad power factor results in wastage of the exchange.

In order to transmit the same active power, in a circuit with low PF, higher currents are needed, resulting in greater losses, wires and equipment required.

The aim of this project is to design a device which could produce as high FP as possible. Sometimes it is not likely to get a good ratio of active power only with normal operation of the device. For this reason, **reactive power is often compensated**. In some cases, an external device or component, such as a capacitance, is added in order to contribute with reactive power to the consumption of the load. This solution is not very precise, and therefore, for optimal compensation, a closed loop control of current is usually employed. Controlling the diphase between current and voltage in the output it is possible to get FP closed to 1.

3.3 Inverter topology

Once the main features of the problem are established, the first issue to consider on the designing is the topology of the inverter. Normally, inverters deal with high voltages and a boost step is required. DC/DC before inversion or AC/AC after it are usual solutions added to the inversion step.

However, the solution adopted in this project will be that of only the **DC/AC conversion,** as the desired voltage levels can be achieved. In case a higher voltage is required, an external boost can be added.

Regarding the bridge topology, a **Full H-Bridge** will be used. This configuration requires two branches and 4 interrupters. As explained in the previous chapter, each elementary switching cell contains a transistor and a diode.

The topology selected (Figure 3.2) implies benefits in regard to Half- Bridge as lower voltages in the semiconductors and lower input voltage for the same output. Switching losses are higher in a Half-Bridge, as semiconductors deal with 2Vdc.

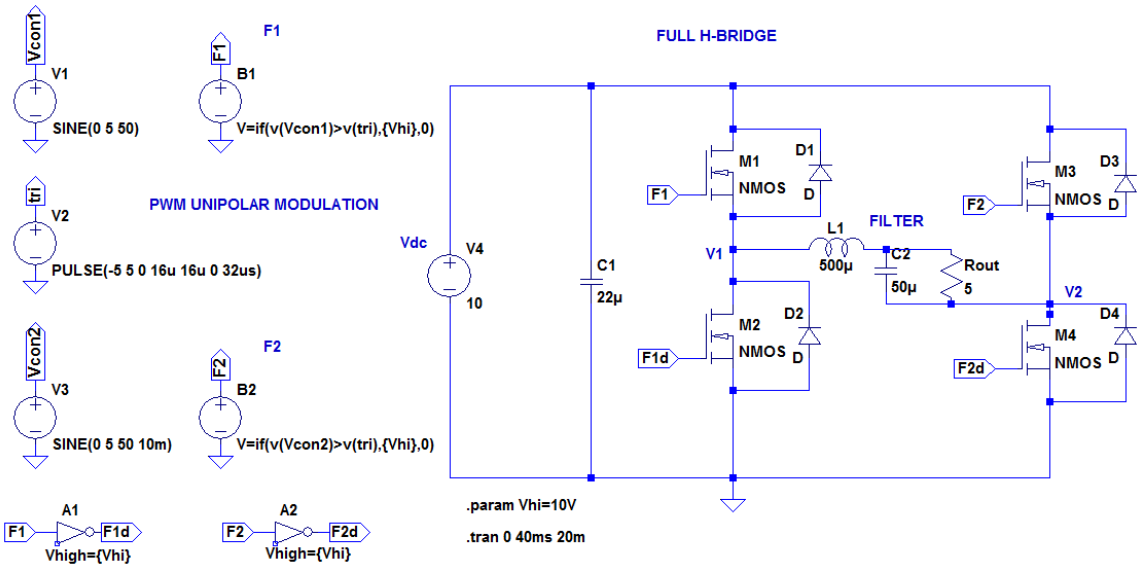


Figure 3.2: Full H-Bridge schematic with LTSpice software.

3.4 PWM Modulation. Switching frequency

A key decision in the design of an inverter is the value of the switching frequency. In the following pages, an analysis of the options will be developed. Also it is necessary to select the modulation to apply in the bridge, as the configuration and the calculation of elements vary.

3.4.1 PWM Modulation

In the previous chapter, an insight into different options for PWM modulation was given. The most common ones are **unipolar** and **bipolar** modulation. In this section, a comparison between them will be made and the most suitable option for this project will be selected.

As a starting point for comparison, in both cases **the output voltage** is measured (Figure 3.3), obtaining different values and frequencies. Establishing the same switching frequency for both modulations, in bipolar modulation the **output frequency** remains the same, while in unipolar modulation it **doubles** the initial value. This allows reducing the size of passive components as inductances and capacitors but leads to higher losses. Regarding the **output values**, on bipolar modulation, they oscillate between two values, Vin and –Vin (being Vin the DC input voltage), whereas that on unipolar the output are three levelled (Vin, 0, -Vin). The higher the number of levels for modulation, the better is the waveform.

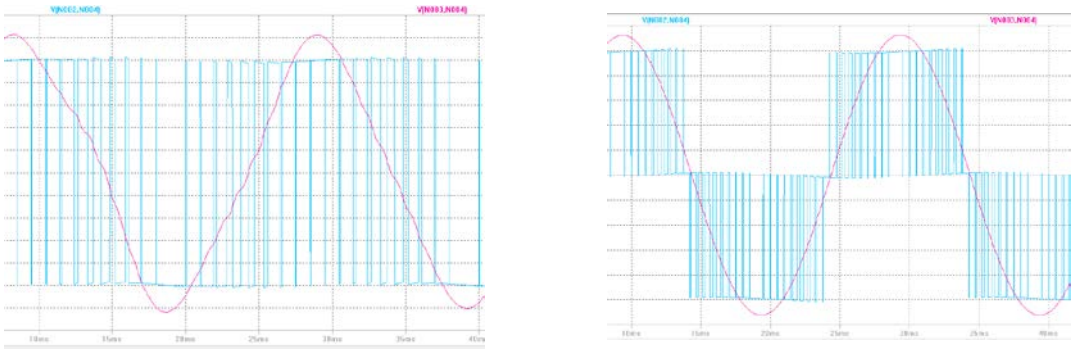


Figure 3.3: Output waveform with bipolar (left) and unipolar (right) modulation.

Regarding **harmonics**, some important points should be stressed. In **bipolar** modulation the output voltage harmonics appear at the frequency of commutation and have twice the amplitude of the branch voltage harmonics. However, in **unipolar** modulation, their frequency is twice the switching frequency and their amplitude is the same as in the branch voltage harmonics. The harmonics of the odd families do not appear in the output, as in every branch they are in phase. As the higher harmonics (first family) are odd and the frequency of the rest is higher than in bipolar, they are easier to filter. This has a direct impact on the **filtering inductance**, resulting it four times lower in unipolar than in bipolar. Another way of reducing harmonics is elevating

the switching frequency, and therefore easing their filtering. However, with this solution the losses can become really high.

Taking into account the criteria described at the beginning of this chapter, the **efficiency** achieved with each modulation should also be analysed. Setting the **same output frequency**, on bipolar modulation the switching frequency is two times higher than in unipolar. For this reason, the switching losses are higher. An insight into Figure 3.4 lets see the different efficiencies between unipolar and bipolar modulation regarding power and with a switching frequency of 20 kHz. The maximum rates are achieved with a unipolar modulated H-Bridge, whereas the Half-Bridge configuration gets the worst results.

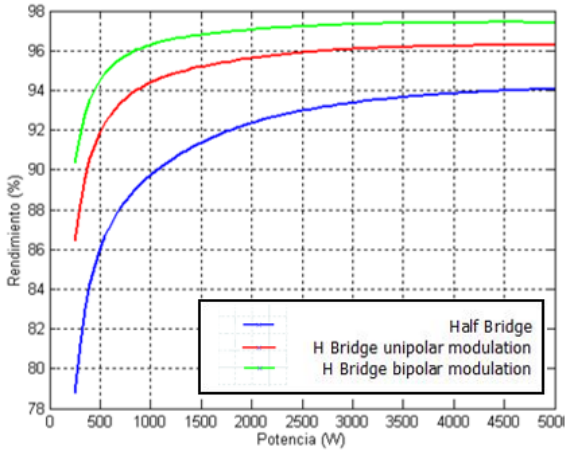


Figure 3.4: Efficiency for f=20 kHz vs. Power [1]

In applications such as photovoltaic systems, it is necessary to consider the differences regarding **common mode voltages**, but for the current design they will not be a decisive fact for decision.

After the discussion developed it is decided to implement **unipolar modulation**. **Efficiency** is the key feature for this decision. Also the ease to filter harmonics and the possibility to have twice the switching frequency in the output are pros that influence on the selection. As the circuit is made for educational purposes, the **size** of the components is wished to be as small as possible, and the output inductance can be reduced with this modulation.

3.4.2 Switching frequency.

One of the most important aspects in the design of an inverter is the selection of the switching frequency, as **losses** and the **quality of the waveform** are directly related to its value. As in this case the generation of the PWM is digital, the switching frequency will be set by the **microcontroller**. A complete discussion of the possibilities will be developed in Chapter 4. Also different options will be tested on the real circuit, being the results collected in Chapter 5.

3.5 Selection of semiconductors

Taking into account the information given in Chapter 2 about semiconductors, in this section a further comparison between some of them will be given, in order to explain the design decision.

The first semiconductor commonly used in power electronics was **BJT**. Its simple structure and capacity to amplify current are two of its main strong points. However, in order to be turned on, high base current are needed. Also, its turn-off is relatively slow and it is highly susceptible to thermal runaway, due to its negative temperature coefficient.

When **MOSFETs** where developed, they became a good substitute for BJTs. As they are controlled by voltage, the problem with high base current is solved. Also, their temperature coefficient is positive, preventing thermal runaway. They are quick switching devices, and on their turn-off there is almost any current tail. This effect is a consequence of the remaining charges on the PN junction, and affects negatively the turning off of the switcher, delaying it and augmenting losses as a result. Structurally, MOSFETs have a body-drain diode, which can be used as free-wheeling diode for inductive loads. On the other hand, their switching losses are quite noticeable and they cannot deal with very high voltages.

IGBTs combine features from BJTs and MOSFETs. They are controlled by voltage but the output switching and conduction characteristics are those of BJTs. The turn-off is similar to the MOSFET one, but with a larger current tail. This fact makes their shutdown time and losses higher. Other disadvantages are the negative temperature coefficient and the lack of body-drain diodes. However, some solutions can be adopted in order to attenuate those drawbacks, as adding ultrafast recovery diode co-packed to faster the turn off. Furthermore, actual process technology and device structure have been improved, resulting in better switching characteristics [7].

The selection between MOSFET and IGBT depends on the **application**. Solutions are not unique and the features and requirements given in each application determine the choice. However, as a rough view, in Table 3.1, it can be seen a resume of some remarkable features and values.

Feature	MOSFET	IGBT
Voltage	< 250 V	>1000V
Frequency	> 100 kHz	< 20 kHz
Power	< 3 kW	> 3 kW
Efficiency	Light load	Full load
dV/dt	Limited	High

Table 3.1: Resume of features and selection [12]

Design decision

In this current project, **voltage** is the key feature for deciding. As the requirements for the application are very low, the saturation voltage of the IGBTs is a handicap. Glancing over different catalogues and vendors, it is hard to get an IGBT with $V_{CE(SAT)}$ lower than 1.4 volts. As normally two IGBTs conduct at the same time, this results in nearly 3 volts lost between input and output. This is circa a fifth of the DC voltage source so it cannot be taken as adequate. Considering MOSFETs, their behaviour at conducting can be modelled as a resistor with a value in the range of milliohms. Therefore, the difference between input and output is not so remarkable. About **frequency**, although MOSFET are commonly selected in high frequency applications, they can also deal with low values.

At the time of simulating, it is possible to compare the output waveform with ideal switchers, MOSFET and IGBTs in the following circuit:

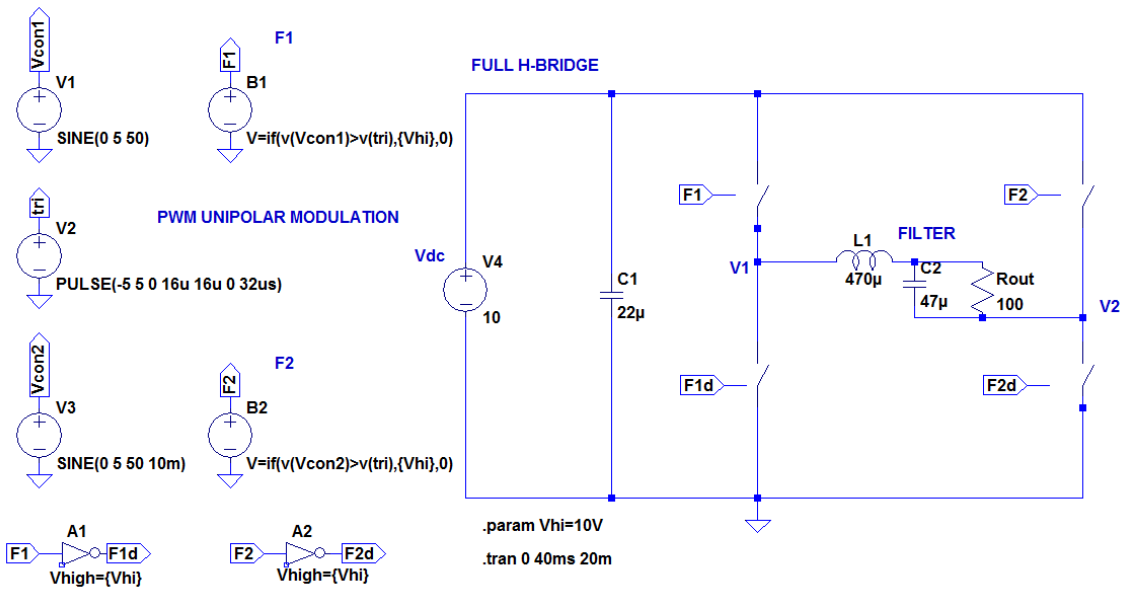


Figure 3.5: Schematic of Full H-Bridge applied for comparison between semiconductors.

For all the cases the input DC voltage is 10V and the switching frequency is set in **10 kHz.** Figure 3.6 shows the different results obtained. The vertical axis division is 2 Volts. The blue waveform corresponds to the output voltage without filtering, whereas the pink one is the voltage in the load.

As it can be seen, while IGBT-bridge remains under 6 volts, MOSFET-bridge is closer to 10 volts. Also, the clarity of the switched waveform is notable in case of MOSFETs, and the unipolar modulation can be well distinguished. On the other hand, with IGBTs the waveform is not so clean and the peaks are more irregular.

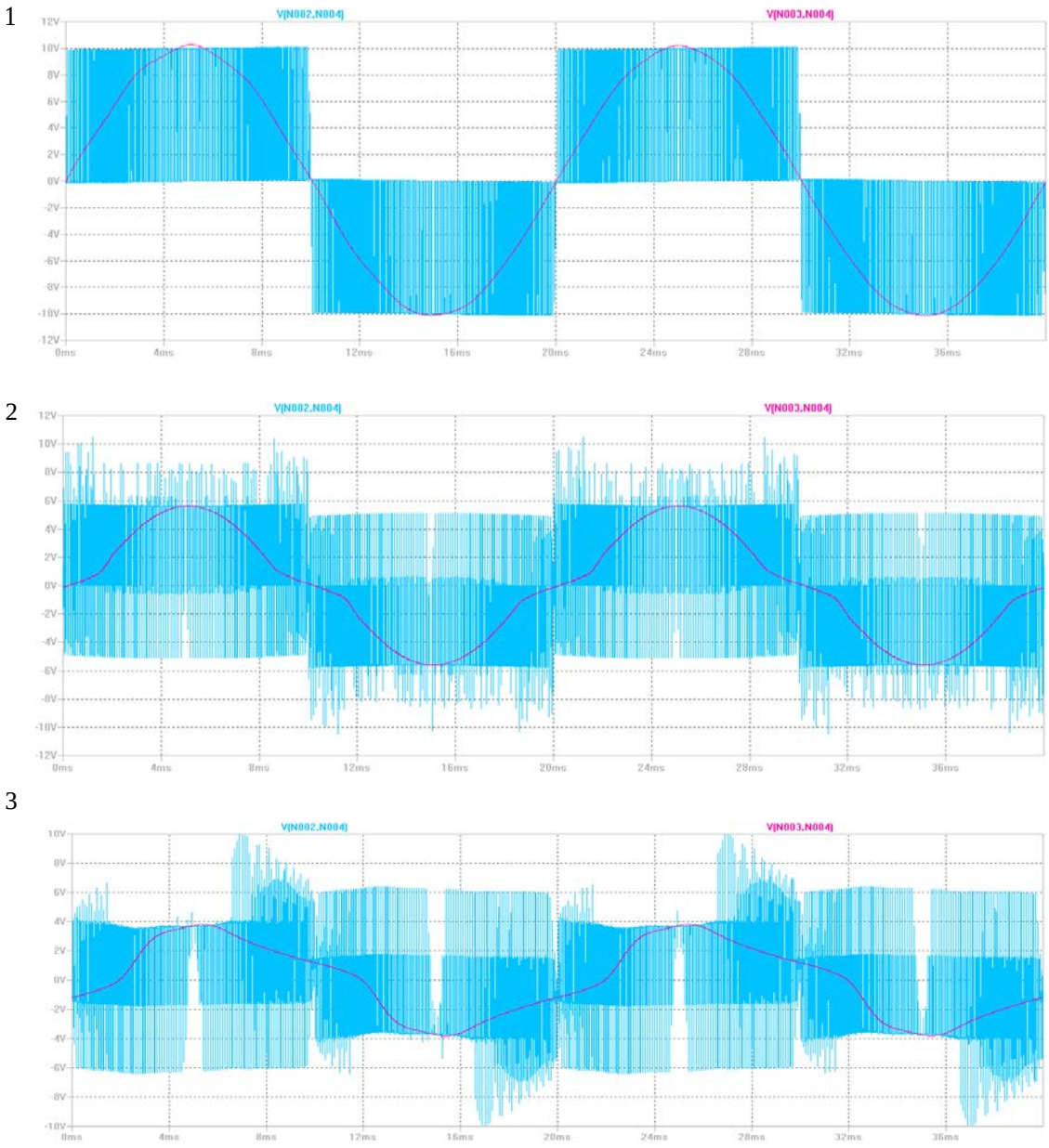


Figure 3.6: Output voltage waveform on ideal switchers (1), MOSFET (2) and IGBT (3).

After the previous discussion, it is finally decided to select **MOSFET** as semiconductors. This way, in order to achieve a three segment switcher, a **MOSFET and a diode in antiparallel** will be chosen.

Regarding MOSFETs configuration, it is possible to find N-Channel and P-Channel semiconductors. As explained before, in power electronics the most widespread type are N-Channel MOSFETs, which have P doped silicon as substrate. When they are turned on, the conducting channel is formed by the movement of electrons. The consequence of employing this type of FETs is that a bootstrap circuit is required to drive the high-side MOSFETs' gates of the bridge.

After consulting different options and vendors available, the MOSFET selected is **IRF60B217**, from Infineon, specially designed for low voltage applications. Its complete features are collected on its datasheet [17]. The package selected is **TO-220AB**, in order to adapt easily to the breadboard and the PCB.

Figure 3.7: IRF60B217 with TO-220AB package

About the **diode**, the specific semiconductor is **1N4001**, a commonly used model, which can be provided by several vendors.

3.6 Dead – times on the semiconductors

Ideally, commutations in the semiconductors are instantaneous. That is to say, when the on order is given to the MOSFET, it turns on immediately. This behaviour allows the two semiconductors on each branch to be complementary in performance, avoiding their overlap.

However, when it comes to reality, the MOSFET turns off a bit later than receiving the order. This time is called **turn-off delay time (td_{off})**. The same case occurs at the time of turning on, resulting in a **turn-on delay time on (td_{on}).** This behaviour is shown in Figure 3.9 of this chapter, being the delay times from t_0 to t_1. Considering the branch as a whole, it would not be a problem in case the td_{off} and the td_{on} of the semiconductors would be equal. Unluckily, in nearly all the semiconductors the turn-off delay time is higher as the turn-on delay time.

As a consequence of this, both switchers conduct during a short period of time, which results in a transient **short-circuit**. Voltage drops and there is a peak of current tending to infinite. Depending on the delay, the value of the current could not reach very high values, as normally the overlapping lasts nano or microseconds.

In order to avoid the short circuits, a fixed delay is introduced on the signal of the semiconductor that will be turned-on. This delay is called **dead time**, and during it the voltage of the branch depends on the current sign. As a result, there is an error in the voltage.

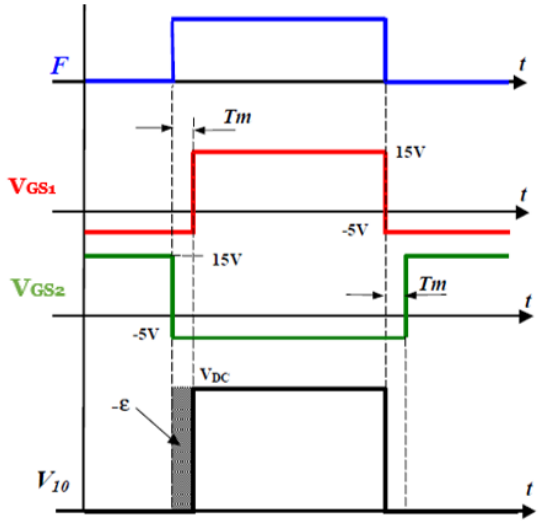


Figure 3.8: Error due to dead time in semiconductors [1]

Figure 3.8 shows the performance of the voltage gate-source (V_{GS}) in both semiconductors of a branch. It can be seen that when the connection function changes its value to high, there is a delay between the switching of both semiconductors (Tm). As a result, an error on the voltage appears (ε).

Nowadays, breakthroughs in power electronics let work with very low turn-off delays. As an example, on the MOSFET selected for this project (IRF60B217), the values for td_{on} and td_{off} are 8.3 and 24 ns respectively.

Also, the **driver** itself helps to handle with this problem. As shown in Figure 3.9, an internal delay is added on the structure of the IR2110 in order to avoid the simultaneous conduction of the switchers.

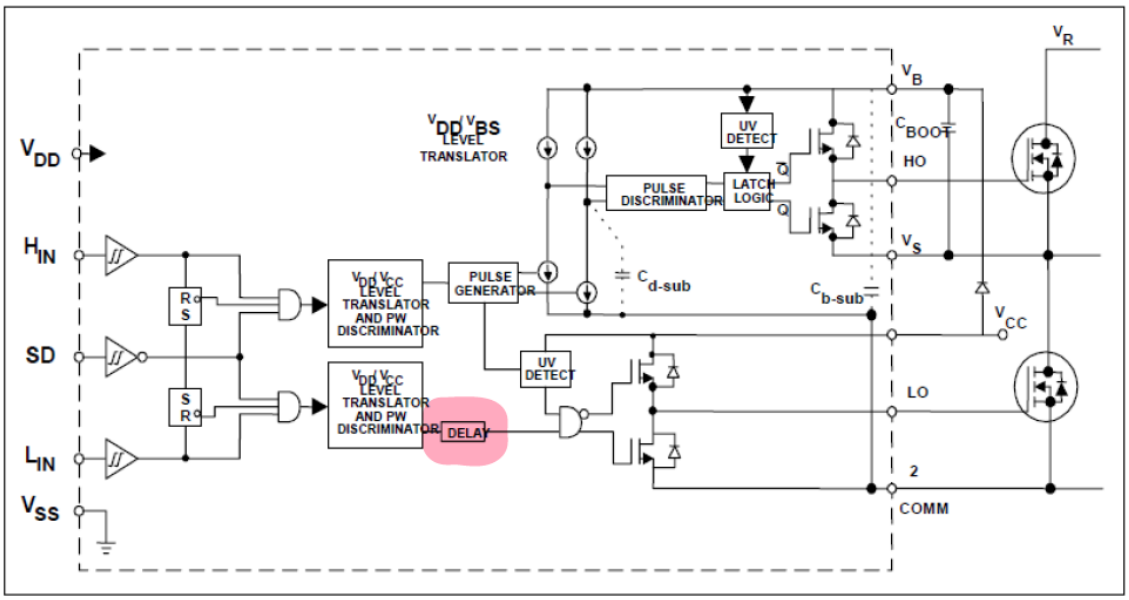


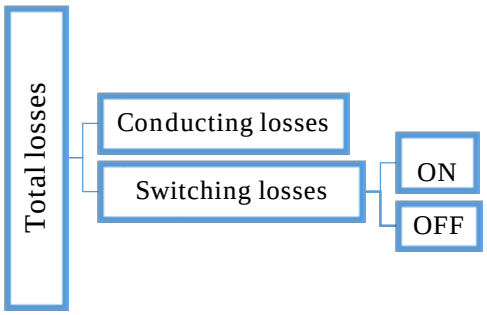
Figure 3.9: Block diagram of the IR2110 [18]

3.7 Losses in semiconductors

An important issue to consider in the design of the inverter are losses in semiconductors. During their performance, semiconductors cause losses not only when switching, but also while conducting. Therefore, an analysis of their internal structure and behaviour should be made in order to compute losses. After this first analysis, in this section an estimation of losses is made. Those computations are required in order to discuss the necessity of a sink.

3.7.1 Analysis

In this current project, the semiconductors selected are MOSFETs and diodes. During their performance, they produce losses that become heat and result in a waste of energy converted. A complete analysis of losses include conducting losses, turn on and turn off losses.



- **Losses in the MOSFET**

When **conducting**, a MOSFET can be modelled as a resistor with value $R_{DS(ON)}$. This resistor normally is of a very low value, in the range of milliohms.

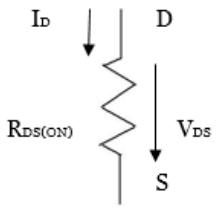


Figure 3.10: Diode conducting model

The **instantaneous power** can be computed as:

$$p\,(t) = R_{DS(ON)}\, {i_D}^2(t)$$

Being i_D the current on the drain. In case i_D is constant, the calculations become easier. This way, the **average power** is:

$$P = \frac{1}{T_{ON}} \int_0^{T_{ON(MOSFET)}} R_{DS(ON)}\, {I_D}^2\, dt = R_{DS(ON)}\, {I_D}^2 \frac{T_{ON(MOSFET)}}{T_{switch}}$$

This expression reflects the computation of the average power only when the MOSFET conducts. For this reason, the duty cycle appears on it. Its value depends mostly on the drain current I_D, as it appears squared.

Regarding **switching losses**, the turning-on and off of the MOSFET should be taken into account, as in both cases voltages and currents appear at the same time, resulting into losses. In figure 3.11, the model of the MOSFET and the variations of current and voltage are shown.

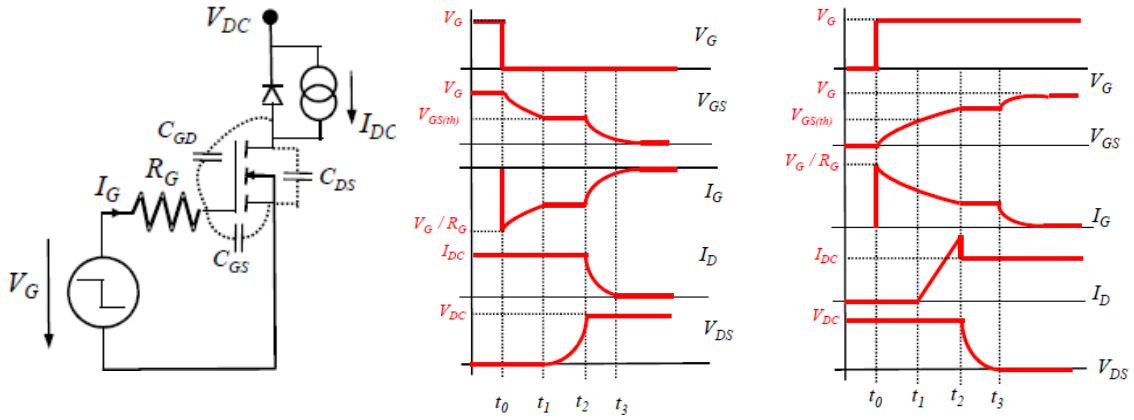
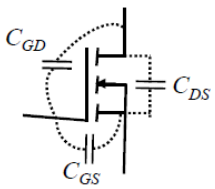


Figure 3.11: Model and waveforms of turning on (left) and off of the MOSFET [1]

Switching losses appear in both cases between t1 and t3. As the switching is not instantaneous, current and voltage deal together during those moments, resulting in losses.

In order to turn-on and off the MOSFET, the electrons of the different layers in which load circulates must be introduced and removed. This behavior can be modelled as **parasitic capacitances.** C_{iss} is the input capacitance, and models the introduction of electrons. C_{oss} is the output capacitance, and takes into account the removal of electrons. Finally, C_{rss} is the Miller capacitance, and reflects the Miller effect (the increase in the equivalent input capacitance of an inverting voltage amplifier as a result of the increase of the effect of capacitance between the input and output terminals). The lower Vds, the higher are these capacitances.



$$C_{iss} = C_{GS} + C_{GD}$$

$$C_{oss} = C_{DS} + C_{GD}$$

$$C_{rss} = C_{GD}$$

Figure 3.12: Capacitances on a MOSFET [1]

Figure 3.12 shows the schematic and composition of the capacitances that can be found on a MOSFET. At the time of computing the losses, the energy involved depends on the drain-source voltage (V_{DS}), the drain current (I_D), the control voltage (V_G), and the gate resistor (R_G).

- **Losses in the diode**

A diode can be modelled as a voltage source and a series resistor when conducting (Figure 3.13). Therefore, a similar procedure as in the MOSFET can be followed to compute the **conducting losses.**

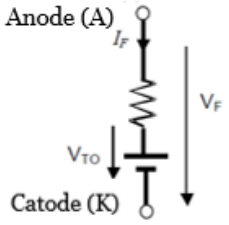


Figure 3.13: Diode conducting model [1]

The instantaneous power is:

$$p = V_{TO} I_F + r_T {I_F}^2$$

Taking Ic as constant, the **average power** is:

$$P = \frac{1}{T_{ON}} \int_0^{T_{ON(diode)}} V_F I_F \, dt = V_F(I_F) I_F \frac{T_{ON(diode)}}{T_{switch}}$$

At the time of computing the **switching losses**, as in the MOSFET, it is necessary to compute the energy lost during the turning-on and off, and multiply it by the switching frequency. However, on a diode, the energy lost during the **turning-on** can be neglected.

In the case of the **turning-off**, it is necessary to relate the value of the graph in the datasheet with the real voltage and current. The energy involved depends on the voltage anode-cathode (V_F), the current (I_F) and the instantaneous change of I_F with time dI_F/dt.

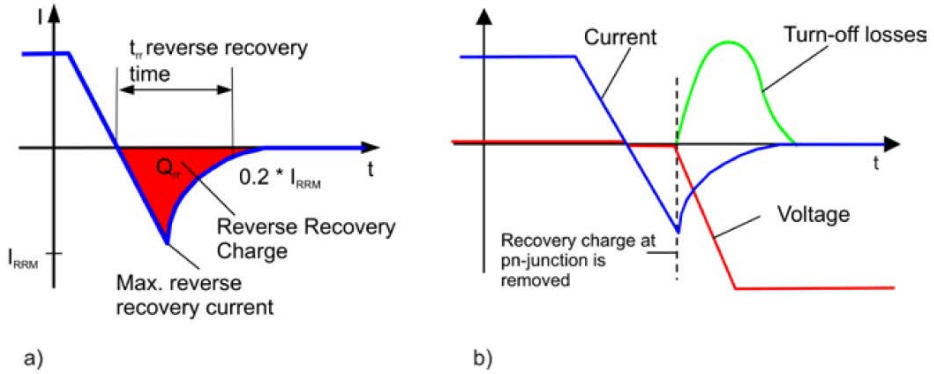


Figure 3.14: Turn-off behaviour (a) and current curve, voltage and switching losses (b) [5]

3.7.2 Calculation

The main difficulty of computing the losses on the inverter is determining the **current** on the semiconductors. As explained in the analysis section, the formulas for conducting losses become simpler taking current as constant. However, on an inverter the current is switched at the switching frequency. For this reason, a method for computing the losses is making the calculations on small periodes of time with different currents. Having those values, it is only necessary to sum them in all the period of time.

As figure 3.15 shows, the current in the branch is **sinusoidal**, with 50 Hz of frequency. The positive semicicle corresponds to the current in the MOSFET, while the diode deals with the negative semicycle. There are several spikes due to transient shortcircuits. The origin of those comes from the dead times on the semiconductors.

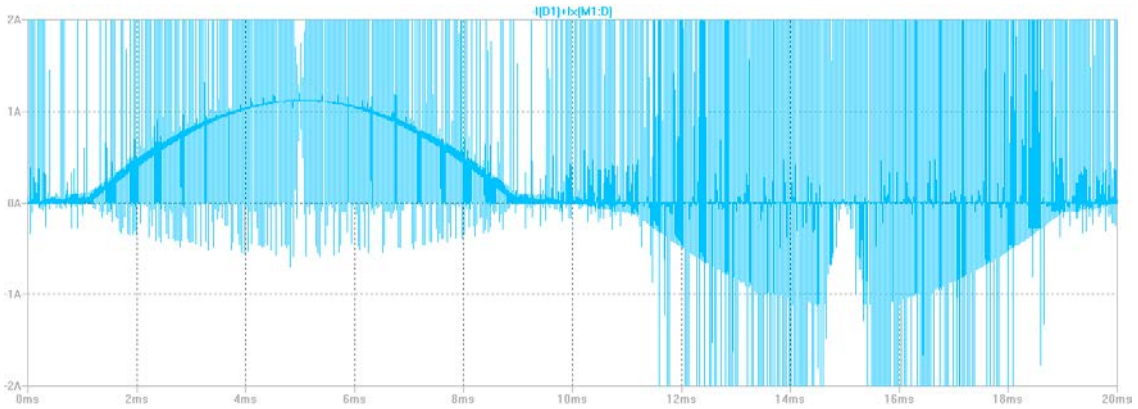


Figure 3.15: Current in a branch in one period (20ms)

The estimation of the losses can be made in several ways. The most accurate is to employ a computation software such as Matlab. An example of code can be found in Appendix 3 of [8].

In this case, it is decided to have a general idea by simulating with the **software LTSpice**. The simulation time is set in 20 ms, in order to have an average value of the losses in one period, and the switching frequency is **31250 Hz**. The main figures are reflected in Table 3.3.

Element		Losses	TOTAL
MOSFET	M1	5.678 W	17.763 W
	M2	3.163 W	
	M3	5.730 W	
	M4	3.191 W	
Diode	D1	167.760 µW	457.918 µW
	D2	62.787 µW	
	D3	167.580 µW	
	D4	62.791 µW	
			17.763 W

Table 3.3: Average losses on the semiconductors in one period

It is highly remarkable the difference between the MOSFETS and the diodes. Also, it can be appreciated that the high-side MOSFETS have nearly twice the losses of the low-side devices.

3.7.3 Power dissipation

As previously commented, it is impossible to obtain a perfect energy transmission between the source and the load. Some of the energy losses its heat capacity and cannot be used, being transformed into **low quality heat**. Not only is the reduction of efficiency a problem, but also the resulting increase of **temperature in the junction**.

The maximum operation temperature on the silicon junction of the MOSFET is **175 °C** (T_{jmax}) [17]. In case higher operating temperatures are achieved, the useful life is reduced. This feature, together with losses on the semiconductor (P_{lost}), the maximum ambient temperature (T_{ambmax}) and the total thermal resistor (R_{thja}), limits the maximum working current on an inverter.

For the discussion in this section, the concept of **thermal resistor** should be clear. On a material, it is defined as its capacity to oppose to heat flow. It is measured in thermal degrees (Celsius or Fahrenheit e.g.) divided by Watts. On an inverter, the total thermal resistor can be divided into several parts, depending on the surfaces in contact.

Regarding thermal analysis, it is possible to model the different junctions and temperatures as an electric circuit. Thus, the power is equivalent to current, thermal resistors to electrical resistors and temperatures to voltages.

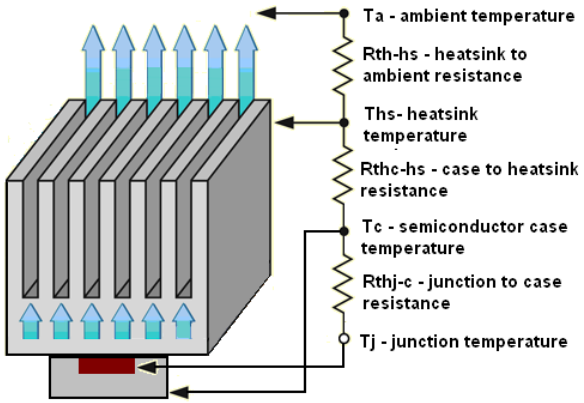


Figure 3.16: Schematic and electric circuit between silicon junction and ambient. [14]

$$T_j = T_a + R_{thja} P_{lost}$$

$$R_{thja} = R_{thjc} + R_{thch} + R_{thha}$$

Power losses can be estimated, and therefore are considered as data on this specific heat analysis. The aim in this section therefore is to discuss the necessity of a sink in order to dissipate heat and the determination of the thermal resistors.

The maximum power losses are

$$P_{lost} = \frac{T_{jmax} - T_{amax}}{(R_{thjc} + R_{thch} + R_{thha})}$$

Therefore, thermal **resistors should be minimized** in order to have greater losses.

For this purpose, some options can be considered. Normally, a **heat sink** is added, allowing better dissipation thanks to its specific surface. In case a sink is not allocated, the circuit is simpler, and the total resistor is only the sum of junction-case and case-ambient resistors.

$$R_{thja} = R_{thjc} + R_{thca}$$

For the actual design, and taking into account the estimations of Appendix A, it is decided to **place a sink attached to every MOSFET.**

3.8 Circuit elements selection

Once the features are established, it is necessary to determine the values for the components of the circuit. Not only is the selection of the semiconductors important. Inductances and capacitors help to damp ripples in voltage and current and therefore must be carefully chosen. In this section, a resume of the elements of the circuit is presented. The complete computations of their value are collected in Appendix A.

3.8.1 Capacitors

At the time of selecting the capacitors, it is first necessary to determine the **optimal value** depending on their role on the circuit. Also, the **type** of the capacitor should be carefully chosen according to the application, as it is desired to have the smallest size possible.

Apart from the **decoupling capacitors** and the **bootstrap components**, which will be further explained, it is important to take into account the **DC Bus capacitor**. The input voltage of the inverter comes from a DC source. However, even the most accurate source produces some noise that distorts the original waveform. Also, it should be considered that on the DC side of an inverter, it appears a pulsating current which frequency is twice the output frequency [1]. This results in a rippling voltage.

For those reasons, it is normal to include a capacitor on the DC bus of the inverter, in order to stabilize voltage. In this design, the element chosen is a **22 µF electrolytic** capacitor. The estimation of this value is collected in Appendix A.

3.8.2 Inductances

On an inverter, as the output voltage is switched, the current is not constant, and oscillates depending on the voltage sign. In order to decrease the ripple current and to attenuate harmonics, an inductance should be placed in the output.

As it is described in Appendix A, the value of the output inductance depends on the PWM modulation selected. In this case, for unipolar modulation and setting a 5% of rippling current in the output, the minimum value of inductance is **360µH.**

Once computed theoretically, the next step is to check by simulating the rippling current. On the MOSFET H-Bridge, the output current waveform on the inductance (on steady state and with $L_{OUT} = 160\ \mu H$) is reflected in Figure 3.17.

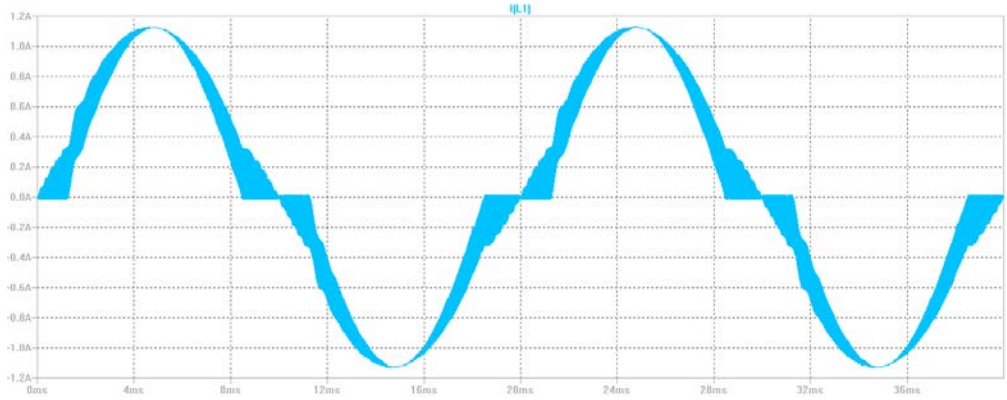


Figure 3.17: Waveform of inductance current with Lout=160 µH.

It can be seen in the figure, that the current ripples mostly on the intermediate values of the sine. It corresponds to the theoretical expectatios, as the maximum rippling value is acquired at a duty cycle of 50% ($m = \frac{1}{2}$), and if it is measured, it corresponds to values around 250 mA.

Depending on the load, the value of the output current can be very low, and therefore it is desired to have as low oscillation as possible. For this purpose, higher inductances can be placed.

For instance, in figure 3.18 an inductance of **470 µH** is employed. As a result, the oscillation of current is only 80 mA. The inductance behaves ideally as a current source, and helps to filter the oscillations. This value will be finally selected.

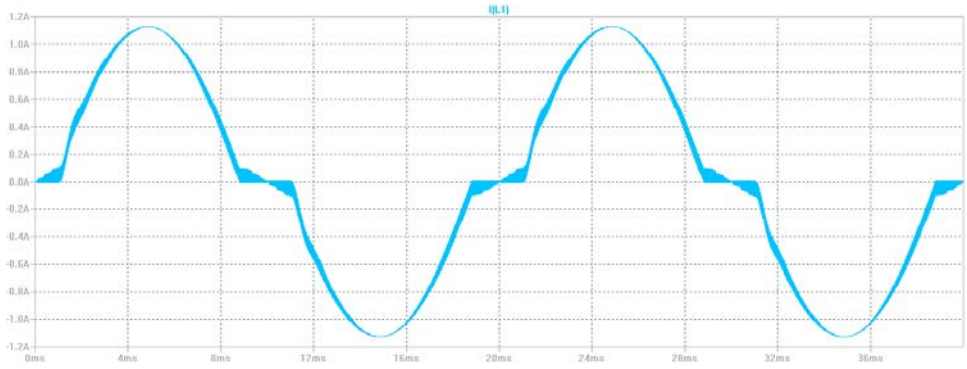


Figure 3.18: Waveform of inductance current with Lout=470 µH.

3.8.3 Filter

The output voltage between two branches (V_{12}) is a **square waveform**. This voltage cannot be applied directly to a load because it could damage it. Therefore, a filter is necessary between the bridge and the load. Not only because of the waveform, but also for **harmonics** is necessary to design and place a low pass filter.

As explained previously in Chapter 2, there are two main types of low pass filters: LC and RC. At the time of selection, it should be taken into account that on an inverter, an inductance is needed in order to filter the ripple current. This inductance can be also used for the filter. Therefore, the best option is an **LC low pass filter**. Moreover, by this choice no extra losses are added, as no resistors are included.

After determining the requisites for cutting frequency, in order not to disturb the normal performance of the device, some computations are made and collected in Appendix A. In conclusion, a low pass LC filter with a series inductance of **470µH** and a parallel capacitance of **47 µF** is selected.

3.8.4 Output load

As explained before in this chapter, depending on the type of load (resistive, inductive or capacitive) the power factor varies.

In order to measure the output for simulation, a **resistor of 180 Ω** is placed in parallel with the filter capacitance. The lower this resistor, the quicker becomes the discharge of the capacitor. On the other hand, a higher resistor value would result in a better waveform. Taking into account losses, the value selected for the resistor allows to have good efficiency. In Chapter 5, a further explanation of the changes in the output voltage depending on the load value can be found. A resume of the components of the circuit is collected in Appendix C.

3.9 Driver

As explained in Chapter 2, drivers are usually employed in converters in order to adapt the connection function to the requirements of the semiconductors. Amplification, protection against low feeding voltage or galvanic isolations are some of the main functions of a driver.

Basically, the gate drive requirements of high-side devices can be summarized to three main points [19]:

- The voltage between gate and source must be from 10 to 15 Volts.
- The gate voltage must be controllable from the logic.
- The overall efficiency should not be affected by the power absorbed by the gate drive circuitry.

Feeding the MOSFET with the right value of voltage is crucial. Not only high values are dangerous for the semiconductor, but also **under voltages**. An example of this is the value of the **resistor** between drain and source in the MOSFET ($\boldsymbol{R_{DS(ON)}}$). The semiconductor can be modelled as a resistor when conducting and therefore its value should be known. However, sometimes it is not easy to compute, as it depends on the value of V_{GS} (gate-source voltage). In the saturation region, the higher the value of V_{GS}, the lower is $R_{DS(ON)}$. The explanation for this involves the internal structure of the MOSFET and the electrons channel that appears when the device conducts. Lower values of $R_{DS(ON)}$ are advisable in order to obtain better efficiencies. Nevertheless, it should be constant to avoid instabilities, as it reflects the movement of electrons in the channel.

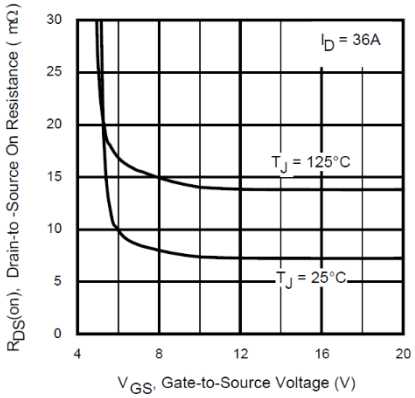


Figure 3.19 shows this variation of values for the selected MOSFET in the design (IRF60B217). Thus, it is recommendable to obtain a V_{GS} higher than 8 Volts. In this particular case, the range of resistance is milliohms, so the conduction losses are not excessively large.

Figure 3.19: On – Resistance vs. Gate Voltage [17]

The connection function applied to the semiconductors, without including a driver, comes from the microcontroller. The PWM output of the **Arduino UNO** oscillates between **0 and 5 Volts**. From Figure 3.19, it can be seen that the resistor's value is not constant in that range. This means, in case of a small oscillation in the PWM output value, the resistance changes considerably.

To avoid this, a driver which provides at least 8 Volts is needed. The drain-to-source resistor is around 7 mΩ for those values. Furthermore, the input gate values in the MOSFET should be over the threshold value, otherwise its **saturation** is not achieved, and conducting losses are excessively high.

For these reasons, a driver is placed between the control and the semiconductors. The model selected is **IR2110**, by International Rectifier-Infineon Technologies. As explained on its datasheet [18], it is a high voltage, high speed MOSFET and IGBT driver with independent high and low side output channels. It is adapted to switch the high-side MOSFETs with a simple external bootstrap circuit. Also, propagation delays are matched and its internal structure assures a minimum delay between internal signals in order to reduce the dead times on semiconductors.

Figure 3.20 shows the typical connection for the device, set from its datasheet.

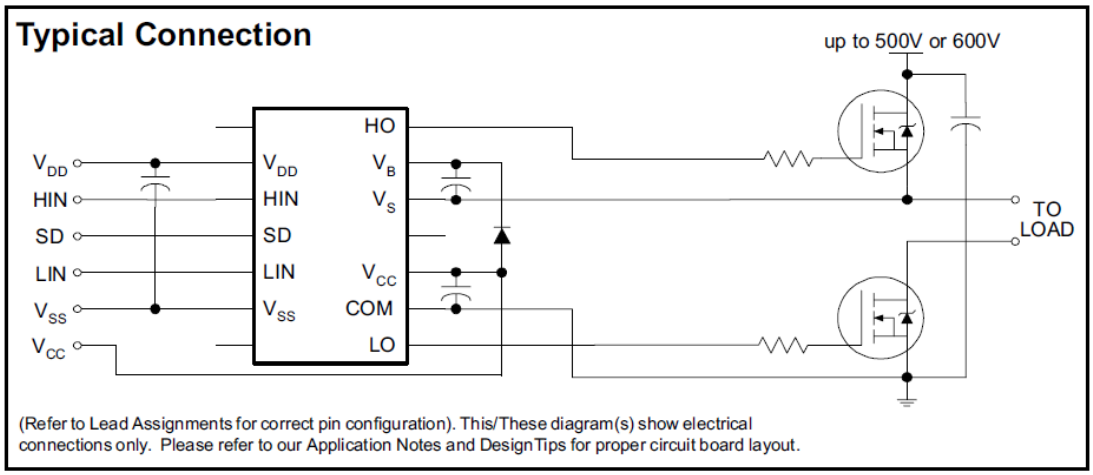


Figure 3.20: Typical connection of IR2110 [18]

In order to assure a safe operation, it is necessary to define the **input voltages**, as well as the **external elements** required. Strictly, only a **bootstrap** diode and a capacitor between V_B and Vs must be placed with the aim of feeding the high-side gate drive circuitry of the driver. However, it is recommended to place **decoupling capacitors** on the Vcc and V_{DD} supplies to compensate inductances in the supply lines [19].

3.9.1 Input voltages

The IR2110 driver requires two different levels of feeding voltage. On the one hand, V_{DD} is related to the input logic part, and must be adapted to its level. On the other hand, V_{CC} deals with the output voltage, and therefore should be chosen according to the desired levels.

SD is a shutdown input that allows disabling the driver. In this case it will be set into zero, but another example of connection is connecting it by a capacitor to V_{DD} (Figure 30 from the Application Note AN 978 [19]). For its part, V_B is a floating input referred to Vs and corresponds to the triggering of the high-side MOSFETs.

Following the recommendations of the vendor, the values set for the input voltages are collected in Table 3.3. The only value that cannot be defined is Vs, as it is referred to the middle point of the branch of the bridge.

Symbol	Value (V)	Description
V_{DD}	5	Logic supply
HIN	[0-5]	Logic input for high-side gate driver output HO
SD	0	Logic input for shutdown
LIN	[0-5]	Logic input for low-side gate driver output LO
Vss	0	Logic circuit ground
VB	15	High side floating supply voltage
Vs	-	High side floating supply offset voltage. Undefined value.
Vcc	15	Low side fixed supply voltage
COM	0	Low side ground

Table 3.3: Description and values selected for the input voltages.

By powering the driver with **15 Volts,** the output levels are assured to be high enough to feed the MOSFET appropriately. The input return voltages will be set to **zero**, as it is necessary to assure that their values are below the threshold voltage of the MOSFETs in order to turn them off. Regarding the logic input, it is recommended to adapt it to the top value of HIN and LIN. Therefore, it will be fed with **5 Volts**.

In order to achieve this, it is decided to place a **voltage regulator**. Thus, it is only necessary to feed externally the logic circuit with 15V, as the regulator adapts this value to the 5V also required by the driver. The model selected is an **LM7805C** [22], widespread linear voltage regulator available in **TO220 package**, and with a low power consumption. Simply adding two external capacitors, it is possible to obtain 5V of output voltage.

According to its datasheet, the current rating of the logic power supply of the driver is around 350µA for the Vcc input and 20 µA for the V_{DD} input, values low enough to avoid excessive heating. The complete scheme is collected in Figure 3.21.

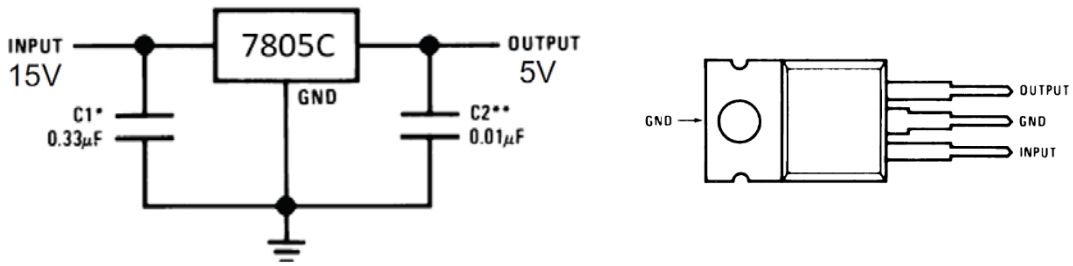


Figure 3.21: Resume schematic of LM7805C.

3.9.2 External elements.

From the **Application Note AN-978** [19], only a bootstrap diode and a capacitor between V_B and V_s are required for a standard PWM application with the IR2110. Also some decoupling capacitors can be placed to compensate the inductance of the supply lines. The complete scheme is shown in Figure 3.22.

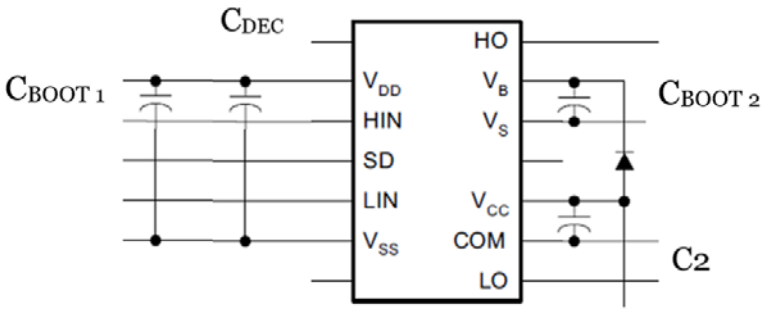
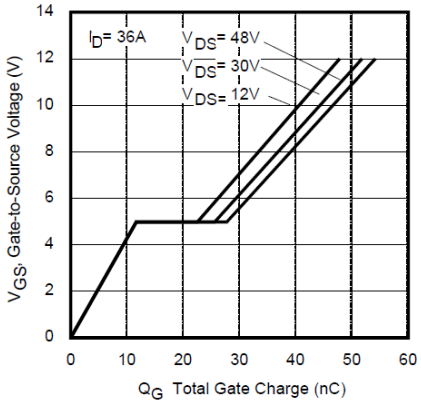


Figure 3.22: Complete scheme of the driver and external elements

- **Bootstrap capacitor 1 ($C_{BOOT\,1}$)**

In the scheme above, two bootstrap capacitors can be found. In the case of **$C_{BOOT\,1}$**, the aim of this component is to give the current peak necessary to turn on the MOSFET. The energy required is stored when the semiconductor is turned-off. Its value should be appropriately computed, as an excessive capacitance involves high charging times. This fact could lead to the limitation of the switching frequency. On the other hand, an insufficient value can lead to voltage drops on the driver's power. As a general idea for its value, the same procedure as in [9] will be followed.

The input supply for the driver should be as constant as possible. Therefore, it can be established a maximum ripple on the capacitor's voltage, being 5% a normal value. In order to have this rippling, the charge of the capacitor (Q_{BOOT}) should be at least 20 times higher than the charge required to turn on the MOSFET (Qg) [9]. As explained before, in case the capacitance is too low, the voltage in the driver's power can drop.



The graph that relates the typical charge gate vs. Gate-to-Source voltage can be found in the MOSFET's datasheet (Figure 3.23). As the drain-to-source voltage is closer to 12 V and V_{GS} will be 10V as maximum, the total charge required to turn-on the MOSFET is **40 nC**.

Figure 3.23: Typical Gate Charge vs. Gate-to-Source Voltage on IRF60b217 [17]

Once this value is defined, the minimum capacity of the bootstrap capacitor can be computed as:

$$Q_{BOOT} = C_{BOOT} * V_{BOOT} = Q_g * 20$$

$$C_{BOOT} = \frac{Qg * 20}{V_{BOOT}} = \frac{40 * 10^{-9} * 20}{10 - 0} = \mathbf{80\ nF}$$

Nevertheless, some additional recommendation given in [19] are also followed. An example given is to increase the bootstrap capacitor value to **470 nF** in order to improve local decoupling and reduce overcharging in case the value of Vs is not high enough. Thus, this value will be selected.

- **Decoupling capacitor (C_{DEC})**

As an extra protection, the placement of a decoupling capacitor is also considered. In case the driver is fed by a switching power supply, some current spikes might be present, and the driver can be negatively affected. In order to avoid this it is possible to place another capacitor. However, the bootstrap capacitor is enough to deal also with the decoupling.

- **Bootstrap diode and capacitor ($C_{BOOST\ 2}$)**

The internal structure of the driver allows the adjustment of the voltage levels for the high-side and the low-side MOSFETs. In the case of the low-side MOSFETS, as they are connected to ground it is only necessary to adapt their gate voltage level to values that assure their saturation. However, in order to achieve the saturation of the high-side MOSFETs, the voltage supplied in V_B must be floating with respect to V_S.

As recommended in the AN-978, a bootstrap capacitor can be placed between **V_B and V_S**, as well as a diode between **Vcc and V_B**. Thus, when the low-side MOSFET is conducting, the capacitor is charged through the diode with Vcc. On the other hand, the diode avoids the discharge of the capacitor when the MOSFET stops conducting [28].

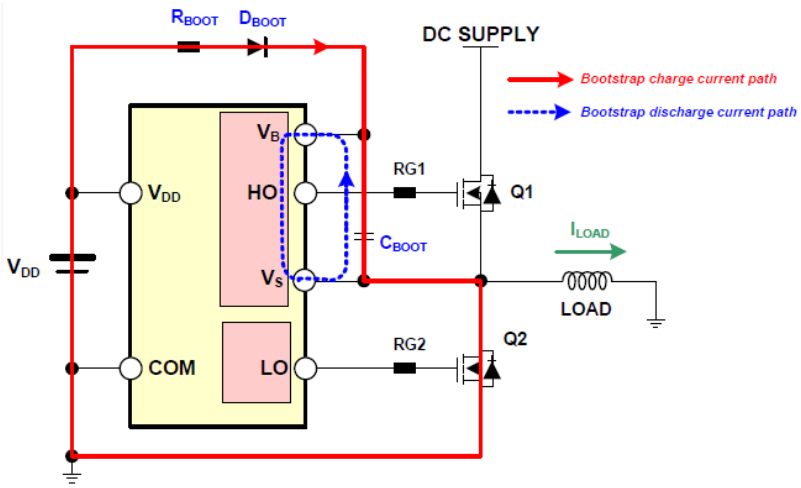


Figure 3.24: Current path on a bootstrap circuit [28]

Figure 3.24 shows the current path for bootstrap charge (red) and discharge (blue) on a similar driver. For the IR2110, V_{DD} in the Figure corresponds to V_{CC}.

The value of the capacity should be enough to assure a stable voltage for the turning-on of the high-side MOSFET. The exact formula for the computation of the minimum value can be found in the Application Note 978 [19]. Following the vendor's advice [19], a value of **470 nF** will be selected. The diode chosen is a **1N4001**, common model used in low voltage circuits.

- **Other elements**

A detailed example of the typical implementation of the driver with an H-Bridge is shown in Figure 3.24. Also, it is recommended to select a value for C2 around ten times higher than C_{BOOST}, as this capacitor supports both the low-side output buffer and the bootstrap recharge. Thus, the value selected is **2.2 µF.**

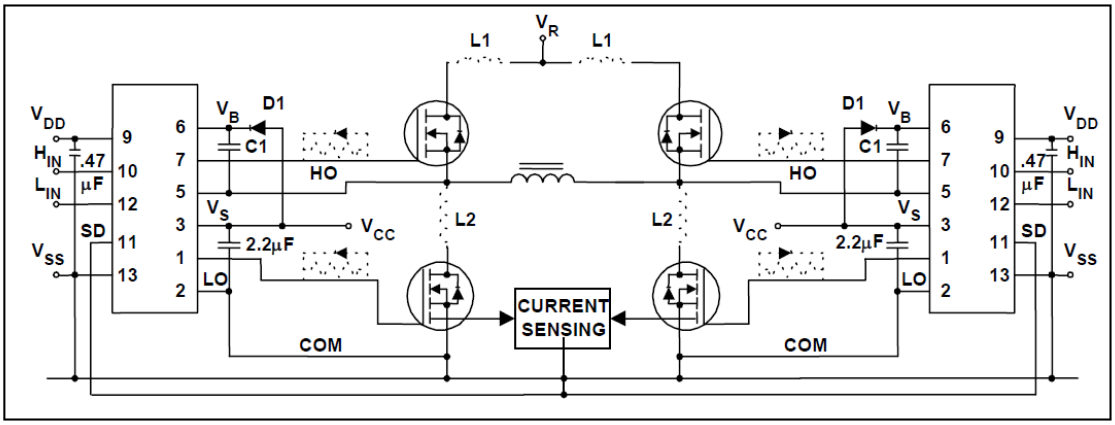


Figure 3.24: Typical implementation of an H-Bridge [19]

3.10 Circuit protections

A very important issue to consider in the design of a circuit are possible problems in operation. The elements of the circuit can be exposed to dangerous operational currents or voltages that may result in their deterioration or even their breakdown. Therefore, some precautions must be taken in order to assure a safe operation of the H-Bridge. In this section, the most important risks will be analyzed. As a solution to them, some proposals will be made.

3.10.1 Overvoltage protection

The PN junction is the base of the semiconductors. If an excessive voltage is applied to it, it can result in its breakdown and therefore, the whole device can be destroyed. On an inverter, one of the most common reasons for overvoltage are the commutations of the semiconductors. When the current is interrupted in an inductive load, the voltage boots. In real circuits, there are parasitic inductances due to the conductors or the connections between elements. When the semiconductors switch, the current changes instantaneously and as a result there are overvoltages. To avoid them a simple solution is to place the elements as close as possible [1].

The **breakdown voltage** varies depending on the specific MOSFET. According to the datasheet of the IRF60B217, the maximum drain-to-source voltage is 60V. As the input source is 10 volts, an overvoltage seems unlikely during normal operation. In the case of **gate-to-source voltage,** its value should not exceed **±20V**. In order to assure those values, a solution commonly taken in this case is to place a **Zener diode** between the gate and the source. This type of diode conducts when a positive voltage is applied between the anode and the cathode. However, unlike other types of diodes, when they are reversely polarized and the 'Zener Voltage' is reached, they act as voltage regulators, as they maintain this value as constant. Thus, they don't allow the MOSFET get into avalanche because of overvoltages. For the purpose of assuring both positive and negative limits, normally two Zener diodes are put joined by their cathodes. In the current design it is decided to place only one **15V Zener diode.** The resulting scheme is shown in Figure 3.25.

3.10.2 Overcurrent protection

One of the most significant sources of danger are excessive currents on the elements. For instance, in case a short-circuit happens, the current will shoot up, only limited by the parasitic resistances in the circuit. However, the values of current achieved could even destruct the device. For this reason, it is common to measure the current in several points of the circuit, in order to disconnect it if the values become dangerous.

- **MOSFET overcurrent protection**

In the case of the actual design, the continuous drain current can be up to **60 amperes**, whereas the device can bear pulses of 225 A. It must be taken into account that the current limits vary with the temperature of the junction. The higher the junction temperature, the lower are the maximum currents allowed in order not to destroy the internal junctions.

As explained previously, in the gate of the MOSFET there is a parasitic capacitance due to its internal structure. Therefore, the switching of the device can be modelled as the charge and discharge of this capacitance. It is desirable to minimise this switching times, as the device deals

with significant voltages and currents, resulting in high losses. However, it is necessary to limit the changes of current with time (di/dt), as they may induce voltage spiking that could damage the MOSFET, especially their gate insulations.

For this reason, it is usual to place a resistor between the driver and the gate, in order to limit the current. The value of this resistor can be determined from the maximum values for output voltage and current in the driver. This device has an output current limit, setting a top value of 2.5 amperes. The maximum voltage that it can provide depends on the maximum input voltage, and it must be below 25 volts. Thus, the gate resistor can be computed as

$$R_G = \frac{V_{out\ driver}\ (\max)}{I_{out\ driver}\ (\max)} = \frac{25}{2{,}5} = \mathbf{10}\ \Omega$$

The **gate resistor** modifies the gate current, affecting the switching of the device. The higher the resistor, the longer the switching takes, and therefore the higher are the losses. However, a minimum value for the resistor is needed in order to limit the current. The MOSFET includes an internal resistor of 2 Ω, but adding an external **resistor** of **10 Ω** poses an extra margin.

In the case of **ideal switchers**, no current flows while they are opened. However, the semiconductor has a very low leakage current in the magnitude of nA. Even though the value is really low, it could charge the gate capacitance of the MOSFET, making it turn on and off and even resulting in its destruction. To avoid this, a **resistor between the gate and the source** can be placed. Its value should be high enough not to affect the normal switching but below a limit determined by the leakage current. A detailed explanation of its computation is given in Appendix A. The standard value chosen for it is **47 kΩ**.

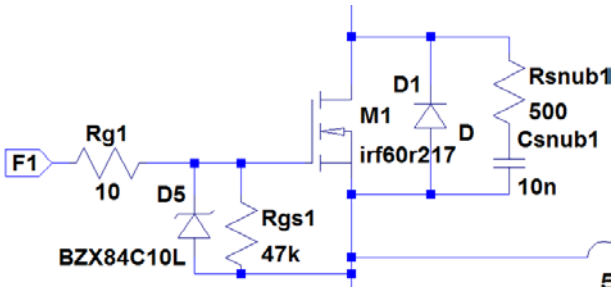


Figure 3.25: Complete scheme of MOSFETs protection

- **Switching protection: snubber circuit.**

As its name indicates, a snubber element is placed to suppress a transient on electrical circuits. In the case of a MOSFET, there are several elements that affect its switching performance, as it can be seen in Figure 3.26.

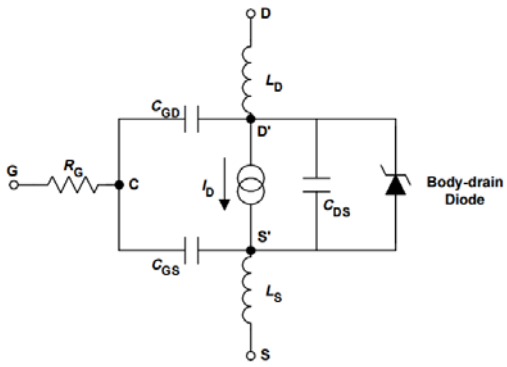


Figure 3.26: Equivalent circuit of a MOSFET showing components with greatest effect on switching [29]

As an example, when the high-side MOSFET turns on, the body-drain diode of the low-side transistor (which was conducting before) turns off. During this process it draws a peak reverse recovery current and snaps off [20]. This phenomenon lasts a short time but has negative effects on the circuit due to **parasitic inductances.** As a consequence of these **reverse current** on the inductances, the voltage is overshot and may result in the turn on of the low side MOSFET again or in avalanche breakdown.

Also the **noise** resulting from commutations can get coupled to the sensitive electronic components in the load, even resulting in malfunctions. For those reasons, it is necessary to follow some recommendations at the time of designing [1], as placing the elements as **close** as possible or locating bypass capacitors next to leads of switching components. Also, the best way to reduce switching noise is to minimise the parasitic inductances [20].

The placement of a **snubber circuit** is a solution that helps to deal with parasitic inductances reduction. An RC snubber is a common combination for protection and it will also be designed and applied in this project. The **capacitor** is placed in order to absorb the inductive energy stored in the load at the moment of switching, whereas the **resistor** helps the damping and makes sure that the capacitance does not discharge instantly on the next commutation.

Following the instructions given in the Application Note AN100-1 form Alpha-Omega [20], some simple computations can be made in order to have an estimation of the optimal values for the circuit. The complete procedure is described in Appendix A. As a resume of it, a **capacitor of 10 nF and a resistor of 500 Ω** will be selected.

The parasitic inductance is distributed in all the PCB and includes the package inductances [20]. They should be minimised to reduce overshoots and ringing. A good layout with the elements as closed as possible is the best way to reduce parasitic inductances. Also, selecting transistors with low inductance packages or which contain integrated Schottky body diodes help to deal with this problem.

- **Load overcurrent protection**

At the time of designing, it is important to define the maximum values for the output. For security reasons, the output current is not desired to be over 5 amperes, as this value can be harmful for humans. From the design features, the output voltage is set to 15 peak volts. Thus, the minimum resistor that can be placed in the output is

$$R \geq \frac{V_{OUT(\max)}}{I_{LOAD(\max)}} = \frac{15}{5} = \mathbf{3}\ \Omega$$

For extra protection, a **fuse** can be placed in the output.

Chapter 4

Microcontroller

One of the main objectives of this project is to implement a digital PWM. Thus, the microcontroller itself is explained in a separate chapter. First of all an introduction collecting **definition** and main **features** is made. After selecting a specific microcontroller type, an insight into the main aspects concerning PWM generation is developed. A key concept are the **Timers** of the controller. Therefore, their definition, operation and parameters are described. Also, an insight into their most interesting Modes of Operation for the current application will be given.

Regarding **PWM generation**, some options available are discussed. The calculation of the duty cycle is also considered, as well as the possibilities regarding frequency.

Although the main code is collected in Appendix B, some excerpts are included in this section in order to clarify the methods described.

4.1 Definition

Microcontrollers can be regarded as integrated circuits which, with a single package, are able to contain a processor core, memory and several programmable input/output peripherals. Commonly used in automatically controlled electronic devices, in this project a microcontroller will also be employed as a response to some requirements for the design.

Nowadays, there is a wide range of microcontrollers available on the market. An example is the Arduino **Board**, which usually contains an Atmel AVR microchip. While other options require a minimum idea of programming, the aim of Arduino is to simplify the process of working with microcontrollers. Its main advantages are:

- Affordable boards and components.
- Simple and clear programming environment.
- Open source and extensible software and hardware. Wide range of tutorials and information available.

Thus, it seems to be the perfect answer for educational purposes, allowing students to take their first steps in programming.

An insight into the different Arduino models reveals a division between entry level and advanced performance boards. For this project an **Arduino UNO** will be selected. Widely employed in basic programming projects, it contains a 16MHz clock, 14 digital input/output pins, 6 analog inputs and 6 PWM outputs. It is based on Atmega 328P microcontroller and can be powered via USB connection or by an external power supply. Being affordable and relatively easy to program, it may be speed- or memory-limited for some applications. However, it fits perfectly to the requirements of the current work.

Figure 4.1: Arduino UNO board

4.2 Timers

Timers are components of the microcontroller that can be used to measure time events. They can be timed by an internal or an external clock source. Due to their importance for programming in the current application, a resume regarding main features and usage will be developed [21].

Arduino UNO has **three hardware Timers/Counters**, which can be configured in several ways to achieve different functionality. They count in synchronism with the main clock. While Timer 0 and Timer 2 are 8 bits, Timer 1 has 16 bits. Thereby, the maximum value they can count up is 255 in the case of 8 bits and 65535 in the case of 16 bits.

In order to clock the timer at the rate desired, **prescalers** can be modified. These are electronic counting circuits that take the basic timer clock frequency and divide it by an integer, according to their configuration [30]. Thus, the overflow period (time required to reach the limit value of the Timer) can be computed as:

$$T_{overflow} = 2^n * \frac{prescaler}{16000000}$$

Being *n* the number of bits of the Timer. The values of the prescaler can be 1,8,32 (only on Timer 1), 64, 128, 256 or 1024. When the Timer reaches its limit value, it restarts counting on the next clock cycle. The default value of the prescaler is 64, which means that normally the overflow occurs every 1.024 ms in case of an 8 bit Timer.

- **Modes of operation**

There are several configurations available for the Timers regarding Operation Modes. These define the main behaviour of the Timer and can be modified by manipulating the values of the registers. A register can be defined as a memory location within the CPU, designed to be quickly accessed in order to have fast data retrieval.

TCCRnA and TCCRnB are the abbreviations of **Time/Counter Control Registers**. They hold the main control bits for the Timer. The output compare pins are defined by the combination of the Waveform Generation Mode (WGM) and the Compare Output Mode (COM) bits. WGM bits control the counting sequence of the counter, the source for maximum counter value and the type of waveform generation that will be used. On the other hand, COM bits have two functions. The Waveform Generator uses them to define the Output Compare state at the next compare match and they also control the Output Compare pin output source [21].

Four main **Modes of Operation** are available in the Atmega 328p, which lead to 7 modes in case of Timer 0 and 2 and 15 modes for Timer 1. The complete descriptions for each Timer are collected in Appendix C.

From the main Modes of Operation, only three are interesting for the current project and will therefore be explained in detail. In order to generalize for all the Timers, the notation employed include an *n* instead of the specific number of the Timer.

Also regarding counters, some concepts should be defined [21]. **BOTTOM** is the lowest value of the counter (zero), whereas **TOP** is the highest value in the counting sequence. The TOP value can be assigned as a fixed value (MAX) or a value stored in the OCRnx Register. The assignment depends on the mode of operation and it defines the resolution of the Timer. The maximum TOP value (**MAX**) is 255 for Timer 0 and 2 and 65535 for Timer 1.

4.2.1 Clear Timer on Compare Match (CTC) Mode

In this case, the **OCRnA** (Output Compare Register) is employed to manipulate the counter resolution, as it defines its top value (TOP). TCNTn (Time Counter Register) increments its value until it reaches OCRnA and then clears to zero. It is possible to generate an interrupt each time the counter value reaches the TOP by using the OCRnA. However, if the OCRnA value is lower than the current value of TCNTn, the counter will miss it. A Timing Diagram of CTC Mode is shown in Figure 4.2.

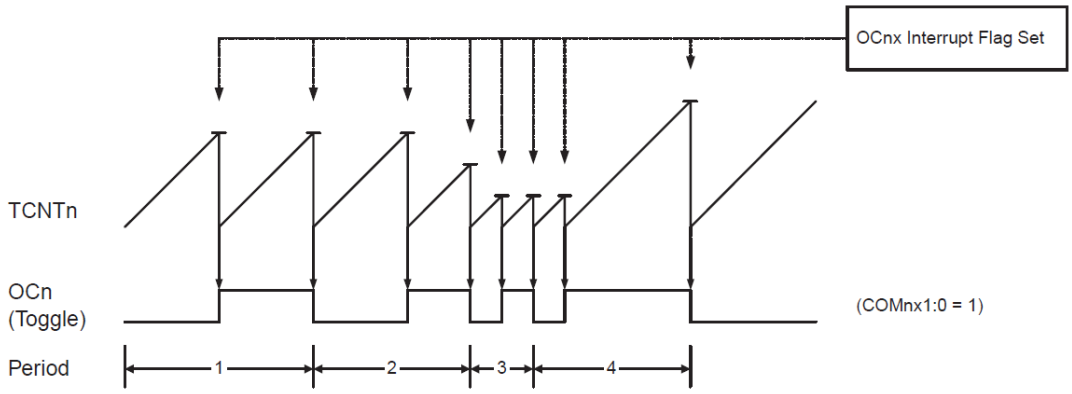


Figure 4.2: CTC Mode timing diagram [21]

4.2.2 Fast PWM

This Mode provides a high frequency Pulse Width Modulation waveform generation option. The counter goes from BOTTOM to TOP and restarts again from BOTTOM in the next timer clock cycle.

Also, it is possible to decide between **inverting** or **non-inverting** mode. In the first case, the output is set to compare match and cleared at BOTTOM, whereas in non-inverting mode the Output Compare is set at BOTTOM after being cleared on the compare match between TCNTn and OCRnx.

Figure 4.3 shows the Timing Diagram of Fast PWM Mode including inverted and non-inverted outputs. The compare matches between OCRnx and TCNTn are represented by the small horizontal line marks on the TCNTn slopes.

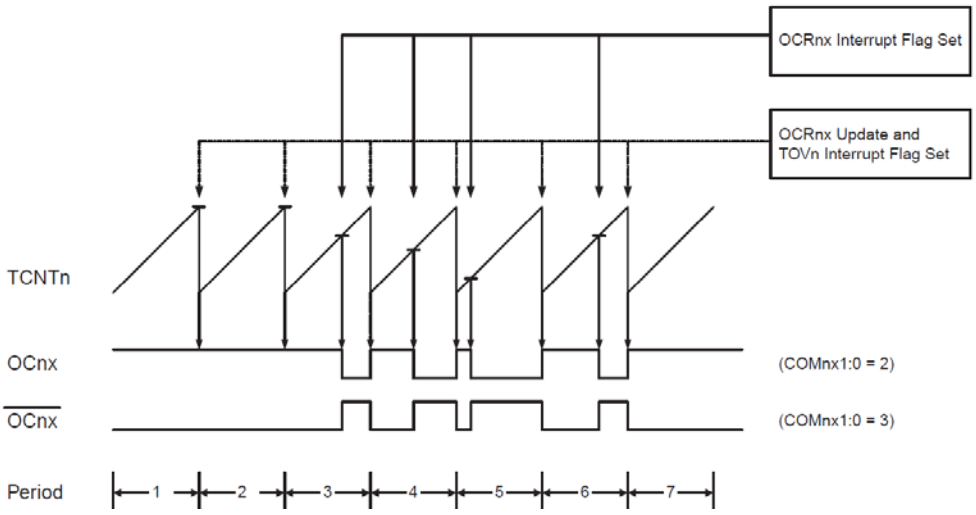


Figure 4.3: Fast PWM Mode timing diagram [22]

4.2.3 Phase correct PWM Mode

This Mode provides a high resolution PWM waveform, and is based on **dual-slope** operation. In other words, the counter goes repeatedly from BOTTOM to TOP and then from TOP to BOTTOM. As in fast PWM Mode, it is possible to have non-inverting and inverting Output Compare Mode. Figure 4.4 shows both Timing Diagrams. As in the previous Mode, the horizontal lines represent the compare matches between OCRnx and TCNTn.

In non-inverting Compare Output mode, the output compare is cleared in the compare match between TCNTn and OCRnx while counting up and set on the compare match while counting down. In inverting Output Compare Mode the operation is inverted.

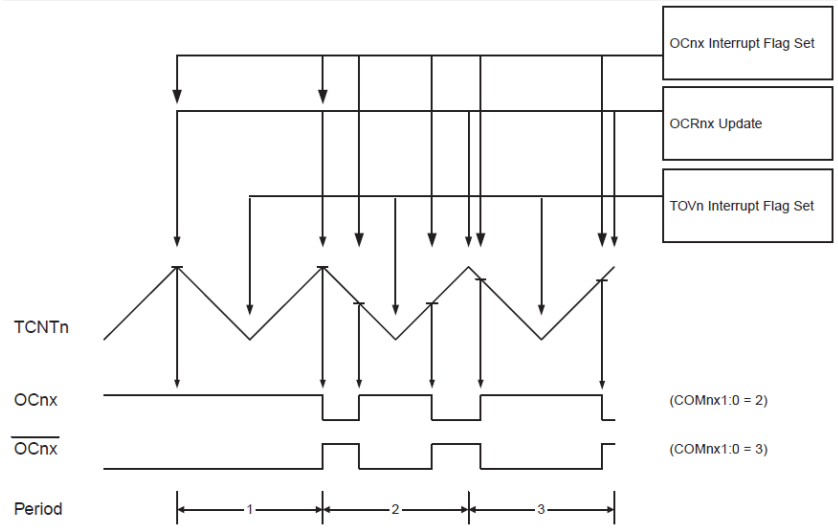


Figure 4.4: Phase correct PWM Mode timing diagram [22]

4.3 PWM

After the theoretical approach made in previous pages, it is necessary to decide how the PMW will be implemented. In **analogical circuitry**, adjusting the triangular waveform frequency to the desired one is enough to define the switching frequency. The duty cycle can be modified by changing the threshold of the control voltage. Figure 4.5 shows the basic schematic and waveforms obtained on an analogic PWM implementation.

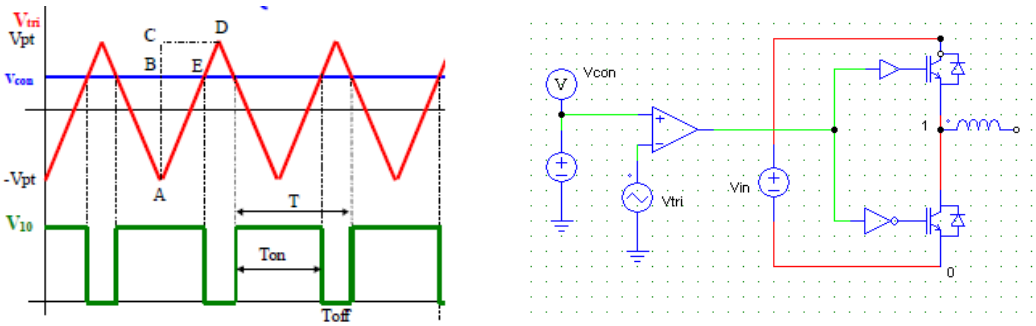


Figure 4.5: PWM waveforms and schematic.

However, this option requires extra circuitry and can also result in extra problems in terms of accuracy due to the signals generation.

Digital PWM implementation allows flexibility and comfort in terms of circuitry. Also, in view of possible design extensions, e.g. to include output waveform control schemes, it is worth to use the microcontroller for generating the PWM. For those reasons, it will be the option chosen. In this section, a complete explanation of how to implement a digital PWM will be made.

To clarify with an **example**, the computations will be made with a resolution of **8 bits**. However, in Appendix B the values for other resolutions can be found.

4.3.1 Calculation of the duty cycle

An important issue to consider is the **calculation of the duty cycle**. As explained in previous pages, in order to obtain a sinusoidal waveform in the output it is necessary to have a varying value for the duty cycle. In analogical implementation, it is enough to set a sinusoidal control voltage.

Going into digital, it is possible to compute the required values for the PWM, which are also sinusoidal. However, it is necessary to define some details in order to get the desired output.

First, it is necessary to take into account the **resolution** of the sine. This value is directly related to the sampling and the PWM frequency. As an example, in case the resolution **is 8 bits**, the values of the dutycycle are in a range from 0 to 255.

The frequency of the output sine is **50 Hertz**, which results in a period of 20 milliseconds. However, in one cycle it is only needed to draw half of the period, as the waveform is symmetric. Thus, the values can be taken advantage for the second half. The 255 steps of the 8 bit resolution can be taken in a semi period, which results in **510 steps for the 20 ms**. Thus, the **sampling period** required to have the 8 bit resolution is:

$$T = \frac{20\ ms}{510\ steps} \cong 39\mu s$$

This results in a frequency of

$$f_{sampling} = \frac{1}{T} = \frac{1}{39\mu s} = \mathbf{25.5\ kHz}$$

Once the sampling frequency is selected, it is also required to adapt the computation to the values that Arduino can provide. As 8 bits are available, this means duty cycle values can be varied in 256 steps.

One option is to compute the sine values in the Arduino and save them to an array. This is possible using the ***sin ( )*** library function, that transform the computations to elementary arithmetic operations. The extract of the code looks like this:

```
1       // VARIABLES DEFINITION
2       int n=510;                      // Number of points in the array for 8 bit resolution.
3       int dutycycle [510];            // Array for the values of the duty cycle.
4       int i=0;                        // Cursor
5       void setup ( ) {
6               for (i = 0; i < n; i++)
7               {
8               dutycycle [i] = (int) ( (1 + sin (2.* PI *i/n ) )* 127.5 +0.5);
9               }
10      }
```

Figure 4.6: Extract code for sine computation.

This way, it is avoided to enter the values manually and some internal memory is saved.

4.3.2 Interrupts

Once the procedure for the computation of the duty cycle is clear, it is necessary to assure that the values are copied to the output pins at the **frequency desired**. Although simpler options can come first into mind, such as using a *delay ( )* instruction to hold the value for the intended time, they are imprecise due to the unknown durations of the remaining instructions in the program.

Interrupts come as a solution to these timing problems. Normally, the program executes its instructions in succession. However, it is possible to set a flag that leads the program to stop the normal operation and execute a specific set of instructions, called **Interrupt Service Routine** (ISR). This function has some limitations, as it cannot have any parameters and should not return anything. The variables shared between ISR and normal functions should be declared “volatile”, to make sure that they are updated correctly.

In an Arduino there are basically two main **types of interrupts**: external and timer overflow. While **external** interrupts are programmed to happen in response to some outside events, **timer overflow** interrupts depend only on the internal clock. Thus, they allow performing a task at specific time intervals regardless of what else is going on the code.

The type of events that can trigger an interrupt are named **interrupt sources or vectors**. When one of this events occur and interrupts are enabled, the code goes to a specific location in program memory (the interrupt vector) [32]. The interrupt vector also prioritizes interrupts and saves them in a queue in case more than one interrupt is waiting to be handled. A complete description

of interrupt vectors can be found in the Table 12.1 of the Atmega 328P datasheet [21]. The list also determines the priority levels of the different interrupts.

In order to make use of **internal interrupts**, it is necessary to configure the **Timers** of the microprocessor. As explained previously, Arduino UNO has three hardware Timers/Counters, which can be set in different ways to achieve different functionality.

Manipulating the values of the **Registers** allows modifying the Mode of Operation of the Timer. In the current work the aim is to have a flag enabling the interrupt when the internal counter reaches a certain value. Consequently, from the modes of operation available, the **Clear Time on Compare Match (CTC) mode** is chosen. Thus, when the Timer reaches the compare match register, the counter is cleared and restarts again from the bottom value.

In order to set the value for the compare match register, OCRA can be manipulated. By defining the **TOP value** for the counter, its **resolution** is also set. Once the frequency desired is defined, the overflow value can be computed as:

$$OCRnx = \frac{f_{clk}}{N \cdot f_s} - 1 \qquad (1)$$

Being *N* the value selected for the prescaler, *fs* the sampling frequency and the frequency of the main clock f_{clk}=16 MHz.

The next issue to think about is how to get the exact frequency desired, as it also depends on the **prescalers.** Following with the example of 8 bits, the sampling frequency is **25.5 kHz**, that is to say the interrupt occurs every 39 µs. The selection of the prescaler is up to the designer, but the OCRnx limits (8 bits for Timer 0 and 2 and 16 bits for Timer 1) should be taken into account. Thus, in case a low frequency is desired, e.g 500 Hz, and no prescaler would be selected, according to equation (1), the value for OCRnx would be 31999, out of the range of Timer 0 and 2. Thus, the minimum prescaler to choose would be 128 for these timers.

In general, high clock rates cause the timer to overflow quicker, as the value for OCRnx is lower. Once OCRnx is below the limit of the Timer, the value chosen for the prescaler is up to the designer.

In the example of 25.5 kHz as sampling frequency, the prescaler chosen will be **8,** resulting in a value for **OCR2A of 78**.

```
1    void setup ( ) {
2    // .....TIMER 2: INTERRUPTION.....
3     cli ( );                          // Deactivating global interruptions
4    TCCR2A=0;                         //Setting the entire registers to zero.
5    TCCR2B=0;
6    TCCR2A |= (1<<WGM21);             // Turn on comparator mode
7    TCCR2B |= (1<<CS21);              // Prescaler at 8
8    // OCR2A = (int) (16000000 / (25500 * 8)) - 1;
9    // Number of pulses it counts until the interruption is activated.
10   OCR2A = 78;
11   TIMSK2 |= (1 << OCIE2A);          // Enable timer compare interrupt
12   sei ( );                          // Global interruption activation
13   }
```

Figure 4.7: Extract code for Timer 2 configuration

Once the timer is configured, it is necessary to devise the content of the **Interrupt Service Routine (ISR),** that is to say, the code that will run every time the interruption is raised. In the current project, the generation of the PWM should be related to this ISR.

4.3.3 PWM generation

There are several options to develop a PWM on an Arduino board. In this section, an explanation of the different methods will be developed, as well as their pros and cons, examples of codes and simulations.

- **digitalWrite ( )**

Regarding the basics of PWM generation, the first approach could be modifying manually the levels of the digital outputs with the instruction ***digitalWrite ( )*** in order to achieve the desired duty cycle. An example of code with this procedure would be:

```
1       void setup ( ) {
2               pinMode (9, OUTPUT)
3       }
4       void loop ( ) {
5               Ton=1000;
6               Toff = (2000-Ton);
7               digitalWrite (9, HIGH);
8               delay (Ton);
9               digitalWrite (9, LOW);
10              delay (Toff);
11      }
```

Figure 4.8: Code for PWM generation with *digitalWrite ( )*

After selecting the specific digital pin as output, it is necessary to establish the values for Ton and Toff. The instruction ***delay ( )*** pauses the program for the amount of time in milliseconds specified as parameter. That is to say, the microprocessor is unable to work on something else during this time.

Interruptions come as a solution to this problem, and as explained previously, the PWM generation should be performed inside the Interrupt Service Routine in order to achieve precise computations. Thus, the problems related to the use of *delay ( )* are avoided. However, the employment of *digitalWrite ( )* involves some **limitations**, as the last value set in the pin lasts until a new invocation is made for the same pin. As a consequence of this, the real duty cycle may not match with the desired one.

- **analogWrite ( )**

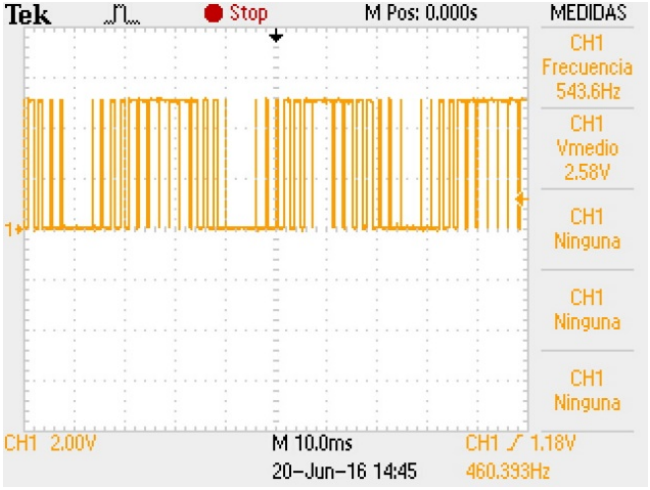
Implementing a PWM with Arduino can be regarded as a technique that employs digital means to produce analog results. A square wave switched between ON (5 Volts) and OFF (0 Volts) can be modified by digital control in order to vary the pulse width and therefore the duty cycle. The instruction ***analogWrite ( )*** allows the generation of a steady square wave with the specified duty cycle until the next call to *analogWrite ( )* or *digitalWrite ( )* on the same pin takes place. It is necessary to specify the pin and the value of the **duty cycle**, which must be between 0 and 255, as the resolution of Arduino UNO is 8 bits.

This function must be applied to the PWM outputs of the board. Arduino UNO has 6 PWM outputs, associated to different Timers and with a default frequency in the output.

PIN	TIMER	Default frequency (Hz)
5	0	490
6		
9	1	980
10		
3	2	490
11		

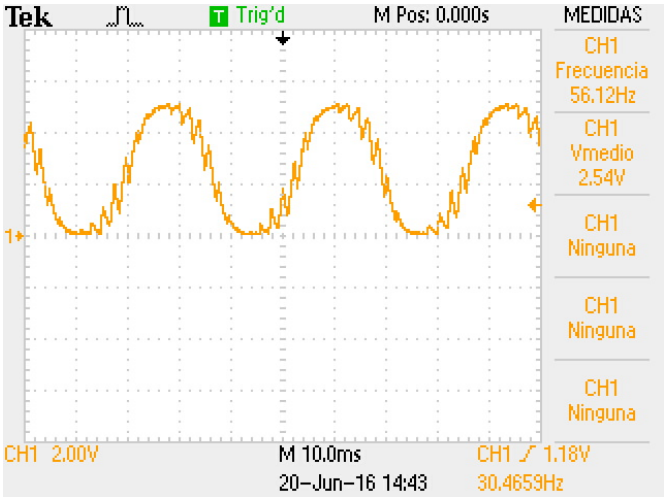
Table 4.1: Pins, Timers and default frequency of PWM outputs on Arduino UNO

Thereby, the next option considered consists of including the instruction *analogWrite ( )* in the Interrupt Service Routine. The output pins employed are 9 and 10.



With an oscilloscope the results are checked and Figure 4.9 shows that an oscillating waveform with varying duty cycle is obtained. However, the switching frequency achieved is 500 Hz, value too low for the current application.

Figure 4.9: Output pin 9 waveform with *analogWrite ( )*



Applying a simple low pass filter with a series inductance and a parallel capacitor is enough to check if the waveform corresponds to a sine. As shown in Figure 4.10, a sinusoidal output is achieved but several spikes also appear.

Figure 4.10: Output pin 9 waveform filtered with *analogWrite ( )* code

- **PWM with Timers**

Although it may require the employment of more internal memory and some skills on programming, it worth most of the time to employ directly the Atmega 328P Timers to create a PWM signal. Thereby, the PWM is directly controlled.

As explained before, there are two Modes of Operation that allow to have a Pulse Width Modulation on the Timers' outputs: fast and phase-correct PWM. One of the key aspects in order to decide which one to select is the **operating frequency**. Due to the single-slope operation, the fast PWM mode can achieve twice the frequency of phase-correct PWM mode, as this last one uses dual-slope operation. This feature makes the **fast PWM** mode suitable for applications such as inverters, in which high frequencies are usually involved.

The maximum frequency that can be achieved with the Arduino Timers is **62.5 kHz**. This figure is more than enough for the current design and allows the employment of small sized external components such as inductances. For those reasons, this mode is selected.

Timer 1 is chosen to provide the **PWM output**, as by having 16 bits, it provides more **frequency control** than the others. An insight into Table 16.4 form the datasheet (attached in Appendix C) reveals the complete Waveform Generation Mode Bit Description. Modes 5, 6, 7, 14 and 15 correspond to fast PWM.

The main difference between them is the **TOP value**, which defines the limit value for the counter and hence its resolution. As explained before, the counter clears itself every time it reaches the TOP value and starts from BOTTOM again. While in case of Mode 15 the TOP value is defined by the Output Compare Register (OCR1A), in the other Modes this register is responsible of the dutycycle. In Mode 14 the resolution is established by the **Input Capture Register (ICR1),** whereas in 5, 6 and 7 it is possible to select directly the number of bits.

The **output frequency** can be computed related to the number of bits set for the resolution (n) as:

$$f_{out} = \frac{f_{clk}}{N*2^n} \tag{2}$$

Being N the prescaler value. Thus, increasing the frequency involves a decrease of the resolution and vice versa.

In order to have more flexibility, **Mode 14** is selected. Therefore, the TOP value can be directly written applying the following formula:

$$ICR1 = \frac{f_{clk}}{N*f_{out}} - 1 \tag{3}$$

Once the requirements are clear, it is necessary to define the bits and registers associated to PWM for Timer 1. The **Timer/Counter Control Registers** hold the main control bits of the Timer. A full explanation on each bit is developed in Section 16.11 of the datasheet [20].

The following code (Figure 4.11) shows an example of how to obtain a fast mode PWM with an output frequency of 62.5 kHz. One of the most remarkable features is that the duty cycle is not constant. Therefore, the **OCR1A** register values are defined by the elements of an array that contains the sine table.

Another issue to consider in the current application is the necessity to obtain **two PWM signals**. As explained before, in order to implement unipolar modulation, in analogical circuitry two control voltages are required, opposited in sign. Thus, this should also be achieved with digital implementation.

Timer 1 is related to pins 9 and 10. While the **OCR1A** register is related to the dutycycle of pin 9, **OCR1B** deals with the correspondent value of the dutycycle in pin 10. The complete code to set the PWM with Timer 1 is:

```
1   .....TIMER 1: PWM GENERATION.....
2   TCNT1= 0;                                   // Initialize counter value to zero.
3   // ICR1 = (int) (16000000 / fs) - 1;        // Overflow value. Prescaler 1.
4   ICR1 = 255;
5   TCCR1A = (1<<COM1A1) | (1<<COM1B1) | (1<<WGM11);
6   // non-inverting, fast PWM. Mode 14.
7   TCCR1B= (1<<WGM13)| (1<<WGM12) | (1<<CS10);      // Prescaler 1.
8   OCR1A= dutycycle [ i ];          // the value for the duty cycle is copied from the array.
9    // OCR1A corresponds to pin 9
10  OCR1B= dutycycle[j];                    // OCR1B corresponds to pin 10.
```

Figure 4.11: Extract code for Timer 1 configuration

Figure 4.12 shows the schematics of the outputs achieved with two dutycycles at the same time. While the upper diagram corresponds to fast PWM mode, the lower shows a phase-correct PWM configuration.

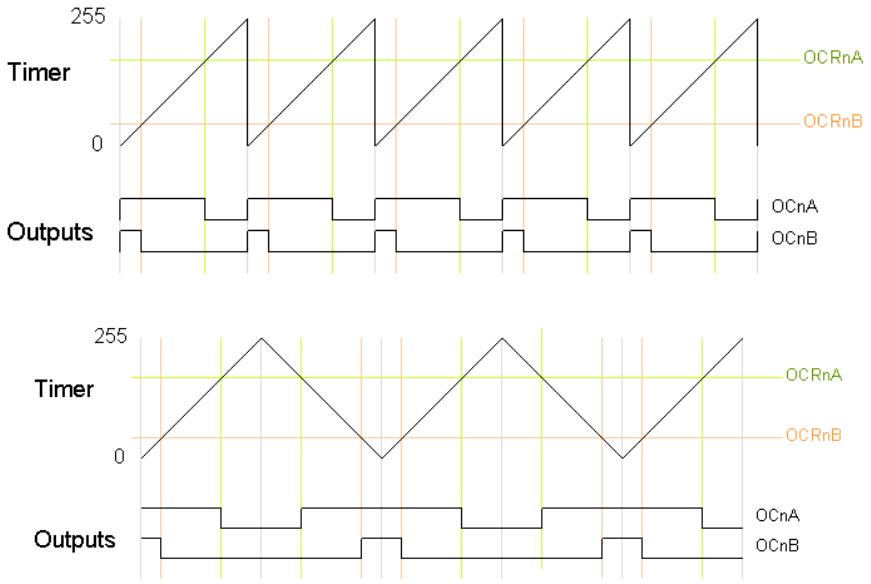


Figure 4.12: Output waveforms for OCRnA and OCRnB with fast PWM (up) and phase-correct PWM (down) [23]

The values for OCR1A and OCR1B have to be refreshed at the frequency desired, which corresponds to the sampling frequency. As explained before, this is possible by including the update in the **Interrupt Service Routine of Timer 2**. Thus, every time the interrupt is raised, in this case **every 39 µs** (defined by OCR2A), a new value from the array is read. The extract code from the ISR is:

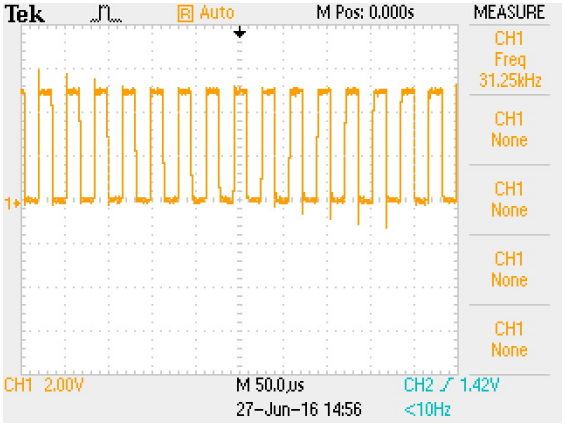
```
1   ISR (TIMER2_COMPA_vect) {
2   // the duty cycle changes taking the values from the array.
3   // It restarts when the cursor overflows n-1.
4           if (i<= n-1)
5           {       OCR1A= dutycycle [i];
6           i++;
7         }
8   else { i=0; } ;
9           if (j<= n-1)
10          {       OCR1B= dutycycle [j];
11          j++;
12        }
13  else { j=0; }
14  }       //ISR Timer 2
```

Figure 4.13: Extract code of Interrupt Service Routine

In this point, the different frequencies that can be found should be clarified. **Sampling frequency** is related to Timer 2, and allows to refresh the value of the dutycycle every time the interruption occurs. As explained before, the value of the sampling frequency for an 8 bit resolution on the sine is **25.5 kHz**. This lets having 510 dutycycle values for a 50 Hz sine.

On the other hand, **PWM frequency** is defined by Timer 1. It is the inverse of the PWM pulse period.

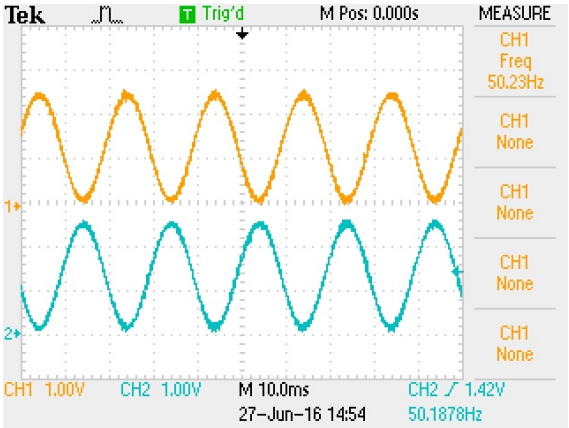
As the frequency for Timer 1 and Timer 2 are not the same, the results vary. In case the PWM frequency is higher than the sampling frequency, there will be **oversampling**. Otherwise some values will be **missed**. An analysis into the different outputs obtained regarding the variation of PWM frequency is made in Chapter 5.



The complete code for a PWM with timers and interrupts is presented in Appendix B. For testing the code, the ICR1 register is set to have a PWM frequency of 31.25 kHz. Figure 4.14 shows how the results obtained correspond to the expectations.

Figure 4.14: Output pin 9 waveform.

As with the previous code, a low pass filter is applied in both outputs, obtaining two inverted sinusoidal waveforms. Thus, the value for the dutycycle is refreshed every time the Interruption occurs.



However, it can also be seen that the amplitude of the filtered sine obtained is below 3 Volts, value much lower than the highest amplitude available (5V). Analysing this problem, it was first considered the option of an excessive filtering, but as the cut-off frequency was 1 kHz and the PWM 31.25 kHz, this idea was dismissed.

Figure 4.15: Output pin 9 and 10 waveform with Code 1 filtered

Another insight into the computation of the dutycycle and the values set as overflow for the Timers revealed that **ICR1** was not adapted to the maximum value of the dutycycle. As the **resolution set for the sine is 8 bits**, the top value of the dutycycle is **255**, and therefore ICR1 should have the same value in order to have the maximum amplitude in the filtered Arduino output.

From equation 3 it is possible to define the **optimal frequency** for this resolution as:

$$ICR1 = \frac{f_{clk}}{N * f_{out}} - 1 \longrightarrow f_{out} = \frac{f_{clk}}{N*(ICR1+1)} = \frac{16.10^6}{1*256} = \mathbf{62500\ Hz}$$

Being 1 the prescaler (N).

ICR1 is related to the resolution of the Timer. It can be modified but it must be clear that the sine table should be also adapted to it. Otherwise the amplitude of the filtered sine in the Arduino output will not be the maximum.

In other words, in case the PWM frequency is set into **31.25 kHz**, ICR1 is 512 (equation 3). If the sine table is not modified and the resolution of the sine is still 8 bits, the maximum dutycycle only reaches half of the range available, as it can vary from 0 to 255. Translated into voltage yields, only around 2 up to 5 peak Volts are achieved in the output. This fact is verified in figure 4.16.

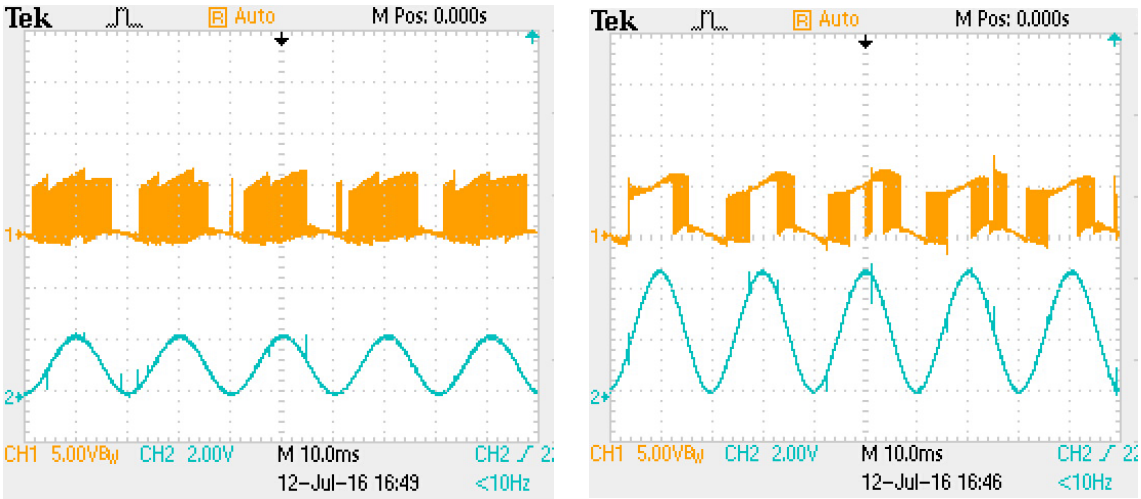


Figure 4. 16: Output waveforms from Arduino with PWM frequencies of 31.25 kHz (left) and 62.5 kHz (right)

As a resume, the approach followed in the code (Appendix B) is taking **PWM frequency** as starting point. Once this value is selected, it is possible to compute the **ICR1** value (TOP of the Timer 1 counter, computed from equation 3). This register defines the resolution and its value should be the same as the maximum value for the dutycycle stored in the array in order to have the highest amplitude for the filtered waveform on the Arduino.

The next figure that can be established is the **length of the array** for the dutycycle (**n**), as double the value of ICR1. This is a consequence of the symmetrical output waveform, and as explained before, the TOP value (ICR1) can take only a semi period of it. As the output signal is desired to be of 50 Hz, it is possible to compute the **sampling frequency** easily as:

$$f_s = \frac{n}{20ms}$$

Being n the length of the array.

From this value, the **OCR2** register (related to the overflow in the interruption) is established as explained in section 4.3.2:

$$OCRnx = \frac{f_{clk}}{N \,.\, f_s} - 1$$

The final code, as well as the values explained for different PWM frequencies are collected in Appendix B.

Chapter 5

Simulation and results

This chapter reflects the results obtained in the project. Before coming to the PCB design, the circuit is tested on a **protoboard** and **simulated**.

The software chosen for the simulation is **LTSpice**, a widespread program that allows the selection of the specific components of the circuit and the simulation of their behaviour.

At the time of working with the real circuit, the procedure followed is gradual. First of all it is decided to check **individual components**, beginning with **the inverter gate** and the **driver**. Then a simple **Half-Bridge** is built to test the driver in a real circuit and finally the **complete H-Bridge** is developed.

An oscilloscope is employed for the display and measurement of the results. Furthermore, some pictures of the circuits built can also be found in these pages.

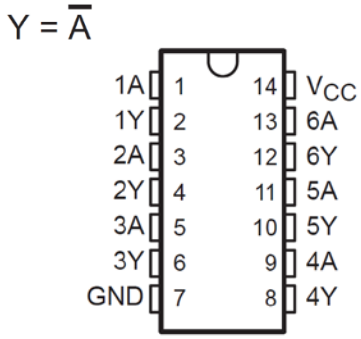
5.1 Testing circuits

To begin with, some simple testing circuits are built. This section collects an explanation of them, illustrated with real pictures and the results obtained in the measurements.

5.1.1 Inverter gate

In order to have **unipolar** modulation, two independent connection functions with their corresponding inversions are required. The microcontroller is programmed to achieve the connection functions and thereby only an additional inverter gate is required to drive the four MOSFETs of the H-Bridge.

The inverter gate selected is a **74LS04**, from Texas Instruments [24]. The pin layout and the internal structure are reflected in Figure 5.1. The supply voltage Vcc is set to 5 Volts, following the vendor's recommendations.



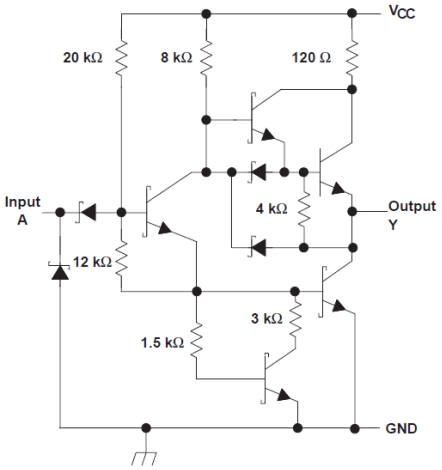


Figure 5.1: Pin layout and internal structure of 74LS04 [24]

The gate selected belongs to **TTL family**. On its internal structure, it has two BJT and a "totem-pole" output. This structure overlaps momentarily when both upper and lower transistors are conducting. As a consequence of this, there is a pulse of current drawn from the power supply that can reduce the noise margin and affect its performance, as well as other parts of the circuit. Therefore it is recommended to add a decoupling capacitor between Vcc and GND in order to absorb the spikes. Its value not only depends on the number of logic inputs activated simultaneously, but also on the frequency of the spikes and the value of the current. Regarding some recommendations [25] and application notes [26], it is decided to place a **100 nF decoupling capacitor.**

A simple circuit for testing is built (Figure 5.2). Output pins 9 and 10 from Arduino are introduced into the logic gate and the results are checked with the oscilloscope.

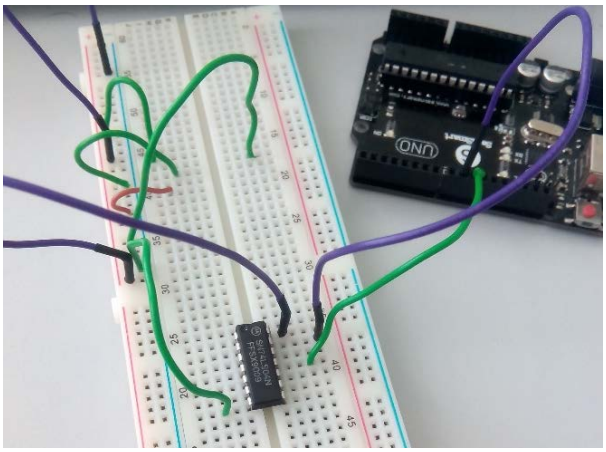


Figure 5.2: Test circuit for the inverter gate

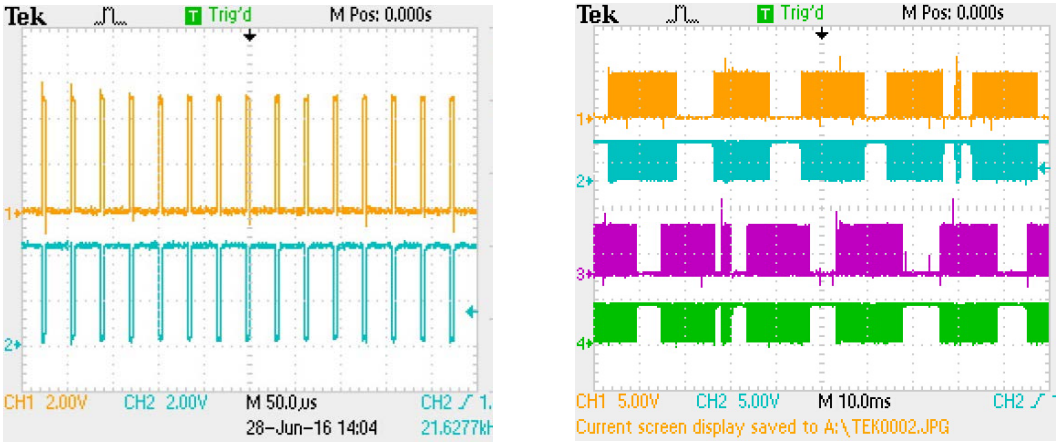


Figure 5.3: Output of the inverter gate with single input (left) and with both PWM inputs

Figure 5.3 shows the performance of the inverter gate. In the left side, the inversion of one input is checked. The full PWM waveforms that will be connected to the driver are shown in the right side.

5.1.2 Driver

As explained in previous chapters, the IR2110 has the aim to adapt the voltage from the microcontroller's output to the requirements of the MOSFETs' gate. Having a look into the recommendations in the datasheet, the following schematic will be tested in order to check the correct behaviour of the driver.

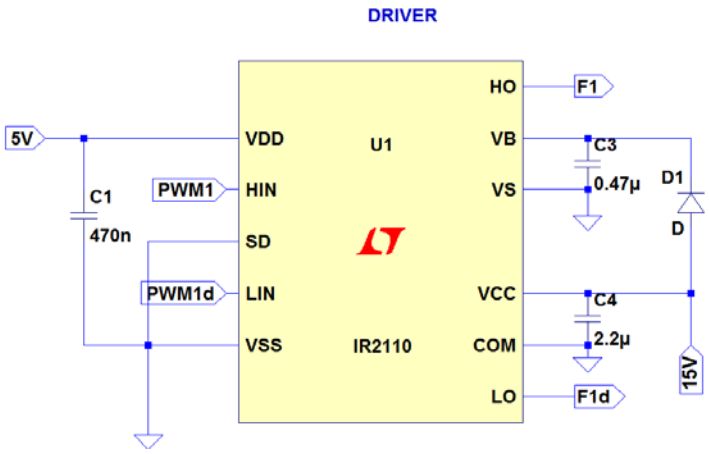


Figure 5.4: Testing schematic for IR 2110, based on [18]

The values selected for the capacitors correspond to the design decisions taken in Chapter 3. Thus, C1 and C3 are **0.47 µF** capacitors and C4 is a **2.2 µF** capacitor.

Regarding the voltages, the logic supply (V_{DD}) is fed with **5 Volts**, as the Arduino's PWM oscillates between 0 and 5 V. This value is achieved with the voltage regulator LM7805C. HIN and LIN are the logic inputs, corresponding to the PWM signal that comes from the inverter gate 74LS04. Vss, SD and COM are set into **ground**.

The low side fixed supply voltage Vcc is connected to a **15V source**. The bootstrap circuit diode-capacitor helps to feed the high-side floating supply voltage V_B, as explained on Chapter 3.

The high-side floating supply offset voltage Vs should be connected to the middle point of the bridge, but as only the driver itself is tested, it will be connected to ground. HO and LO will be measured with the oscilloscope. The testing circuit is shown in Figure 5.5.

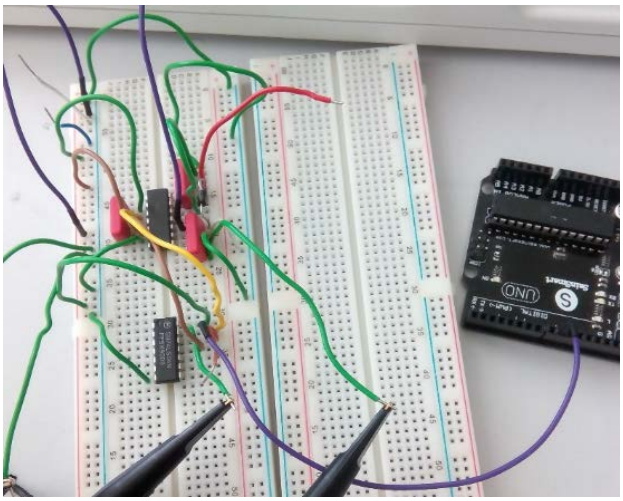


Figure 5.5: Testing circuit for IR 2110

The results are shown in the pictures below. Channel 1 and channel 3 correspond to the inputs HIN and LIN, both coming from the inverter gate. The left figure shows a detail of the performance of the driver, as it can be seen how the input waveforms are amplified to the level desired (in this case 15 volts).

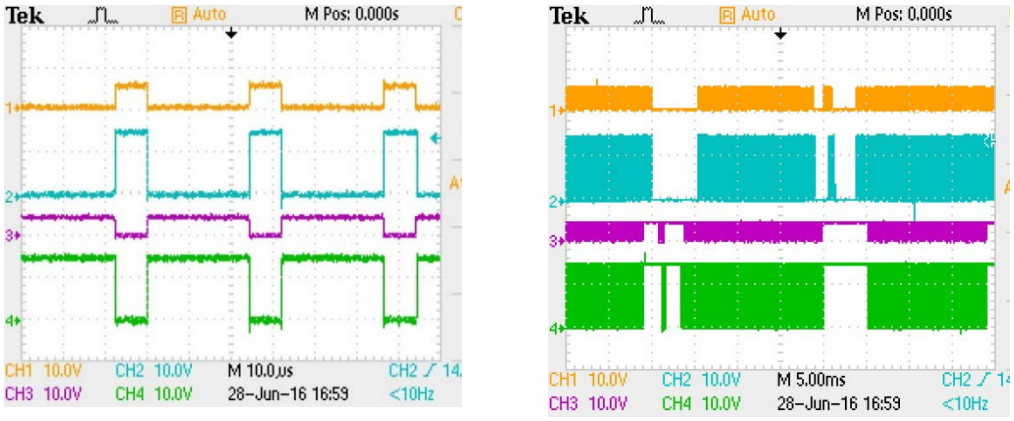
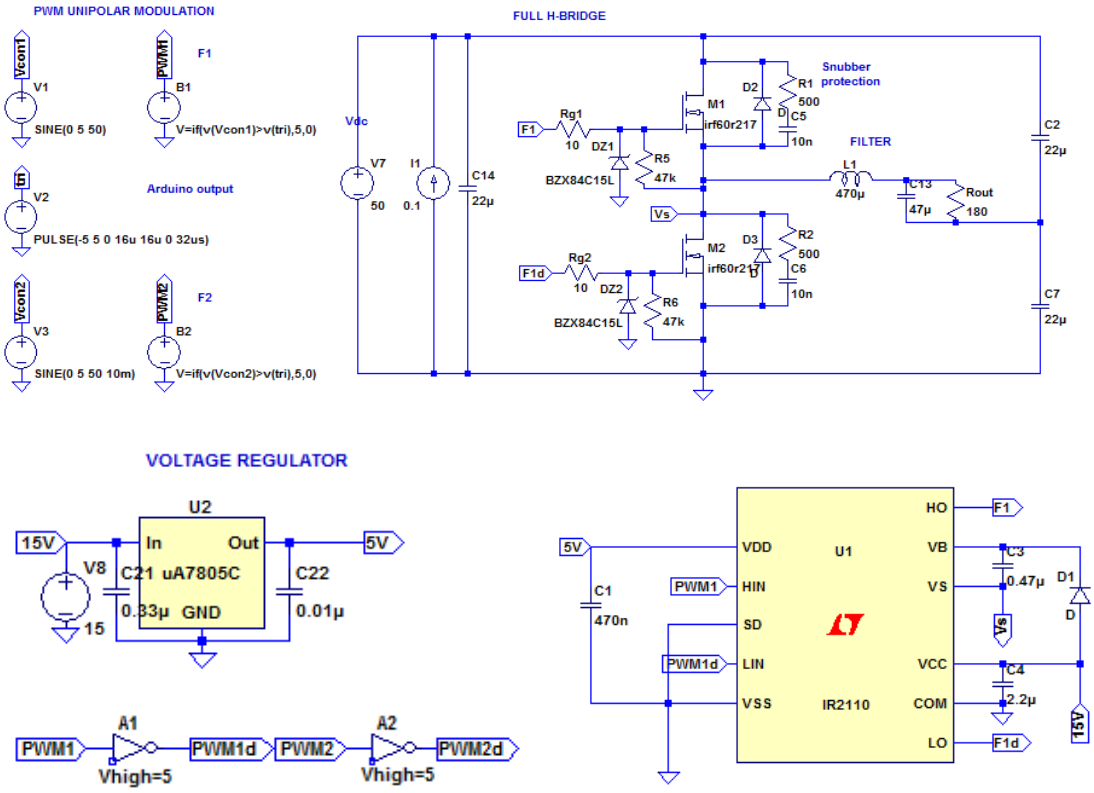
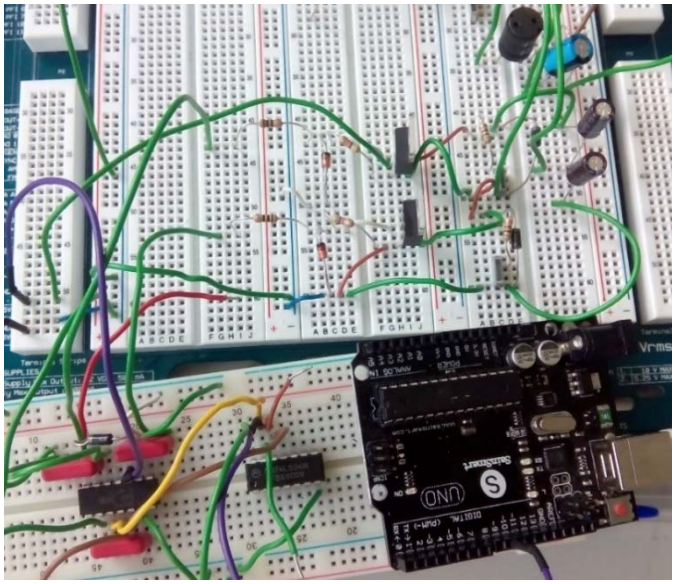


Figure 5.6: IR2110 HIN, HO, LIN and LO pins.

5.1.3 Half Bridge

Once the driver is tested itself, it is decided to check its behaviour on a Half-Bridge configuration. A capacitor voltage divisor is set and the output is referred to the middle point. The load placed is a resistor of 500 Ω and the input voltage is 15V. Regarding the drivers, their Vs pin is connected to the middle point of the leg of the Half Bridge. The complete circuit is shown in Figure 5.7.





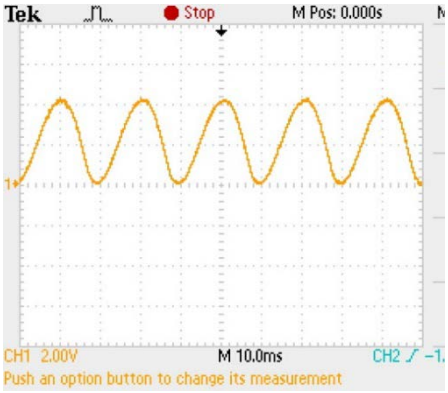


Figure 5.7: Schematic (up), picture and output waveform of half bridge test circuit

The voltage on the output load is measured, and after filtering, its sinusoidal waveform is checked.

5.2 Full H-Bridge

After testing the Half Bridge, the next step is to build and simulate the final circuit. Figure 5.8 shows the complete circuit, which includes the Arduino UNO, an inverter gate 74LS04, two drivers IR2110 and the bridge. The complete schematic can be found in Appendix C.

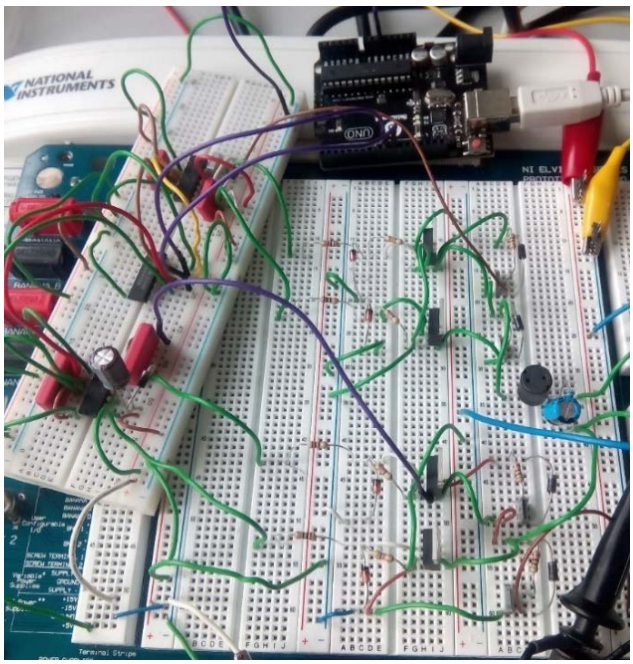


Figure 5.8: H-bridge circuit

In order to analyse the right perfomance of the circuit, the measurements are simulated with **LTSpice** at the same time. For measuring voltages with respect to ground, employing directly the channels of the oscilloscope is enough. However, in case of differencial voltages the procedure applied is connecting two channels of the oscilloscope between the desired points and ground and to substract them with the ***Math*** tool available in the machine.

5.2.1 Vs connection

From the Application Note 978, the Vs pin of the driver should be connected between the source of the high-side MOSFET and the drain of the low-side MOSFET. Thus, as this pin defines the high-side floating supply offset voltage, the gate voltage of the high-side MOSFETs is assured to be between 0 and 15 Volts. In case the Vgs voltage is not enough, the MOSFETs may not saturate and their losses will be higher. The appropriate performance of the driver should be examined in order to assure the saturation of the MOSFETs.

After simulating with LTSpice, the output sine obtained was surprisingly irregular. The simulation was repeated connecting Vs to ground and the quality of the waveform improved. The differences can be seen in the left part of Figure 5.9. The blue waveform corresponds to the output voltage before filtering and the pink one the voltage in the load.

One reason for it could be that while connecting Vs to ground the voltage on the input is fixed, in case it is related to the middle point, the input voltage varies. As a consequence of the small oscillations in the voltage, the software find problems to deal with them and to simulate properly. Also it is necessary to consider the settings of the program at simulating, especially tolerances. The lower the tolerances, the faster the simulation is done. However, excessive low values can affect negatively the results obtained. As an example given, for voltage tolerances of 0.1, only a flat line in zero is obtained in the output.

The output is measured in the real circuit and these imprecisions are not found. The right part of Figure 5.9 corresponds to the waveforms obtained in the real circuit. The orange and green waveforms are the voltages measured on the legs of the output resistor, being the red waveform the difference between them and therefore the output voltage in the load.

On the top of the figure, Vs was connected to ground, whereas the bottom figures correspond to the connection on the middle point of the leg.

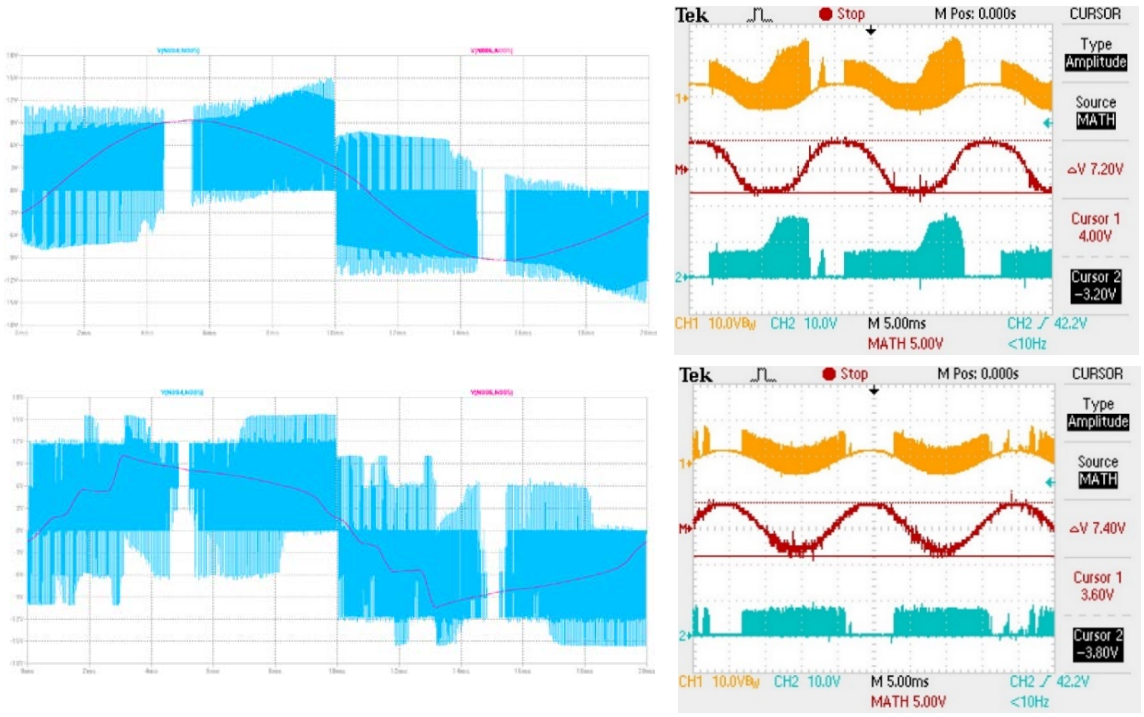


Figure 5.9: Simulated (left) and real output waveform for Vs connected to ground (top) and to the middle point of the bridge (bottom).

5.2.2 Optimal load

Once the output waveform is verified to correspond to the theoretical expectations, it is required to define the details that are not still clear, as the optimal load, the range of PWM frequencies or the best V_{DC} input levels.

To begin with, it is necessary to determine which is the **optimal load** to place, as efficiency and quality of the waveform should be balanced. As the efficiency is inversely related to the load, the maximum value tested is 500 Ω. Otherwise the values of the output current are too low.

The lower the load resistance, the worse is the output waveform. A resistor of 10 Ω is placed and the effects on the quality are remarkable. The left part of Figure 5.10 shows the output waveform with a resistor of 10 Ω, whereas the left side reflects the output for 500 Ω. The output voltage (red waveform) for 10 Ω is almost triangular, in contrast to the sine obtained for 500 Ω.

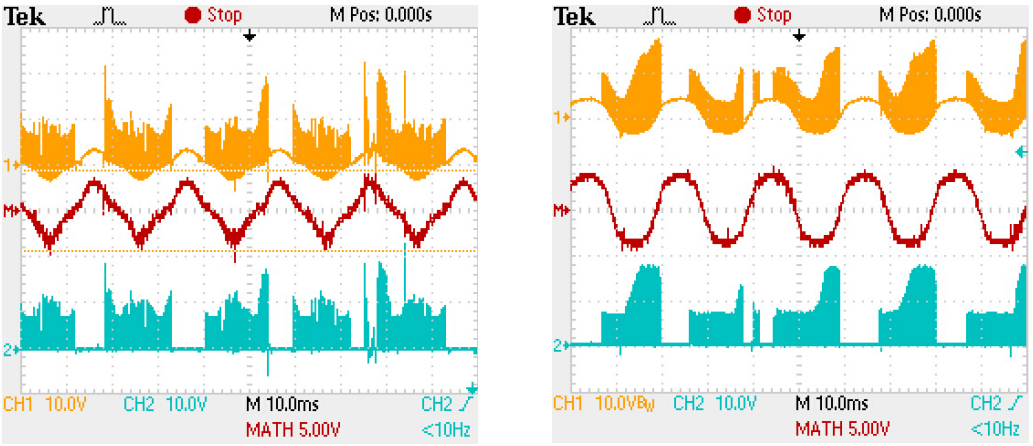


Figure 5.10: Output waveforms with R=10 Ω (left) and R=500 Ω (right)

The test made is with 15 Volts on the DC input and with a frequency of 31.25 kHz. Different resistors are placed, being collected the results in Table 5.1.

R (Ω)	Vout RMS (V)
10	1,35
100	2,55
180	2,70
220	2,77
500	2,77

Table 5.1: Output RMS voltage obtained for different loads.

It can be seen that from values above **180 Ω**, the output voltage remains similar. Therefore, this value will be placed as load.

5.2.3 PWM frequency

As explained in Chapter 4, it is necessary to examine the relation between PWM frequency and the output. In this section, the aim is to analyse the variation of the **Arduino output** in case the PWM frequency varies. To this end, a simple low-pass filter is placed and the measurements are taken before and after it. The filtered waveform should be always 50 Hz.

Following the example of chapter 4 of 8 bits resolution in the sine, in case the PWM frequency is lower than **25.5 kHz** (sampling frequency), some values will be missed and therefore the quality of the waveform will be affected. This is indicated in Figure 5.11. Also, it is possible to observe the peak to peak values in the output for a PWM frequency of 10 kHz (left) and 31.25 kHz (right).

On the other hand, in case the PWM frequency is higher than the sampling frequency, the values for the dutycycle are **oversampled**.

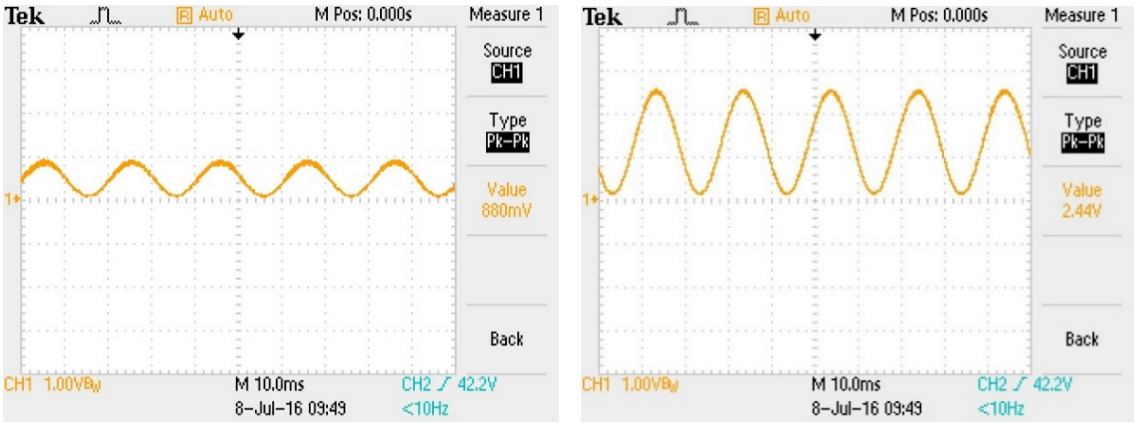


Figure 5.11: Output filtered PWM from the Arduino with 10 kHz (left) and 31250 Hz.

As explained previously, the reduction on the amplitude is due to the necessity of adapt the **ICR1** (TOP value for fast-PWM with Mode 14 of Timer 1) to the highest value of the dutycycle of the array, which is to say to set the same resolution. As the maximum resolution tested for the sine is 8 bits, the best results will be achieved with **62.5 kHz** (see section 4.3.3).

The **output voltage** is directly affected by this variation, being Figure 5.12 an example. The right picture corresponds to the theoretical maximum of the output, with a PWM frequency of 62.5 kHz.

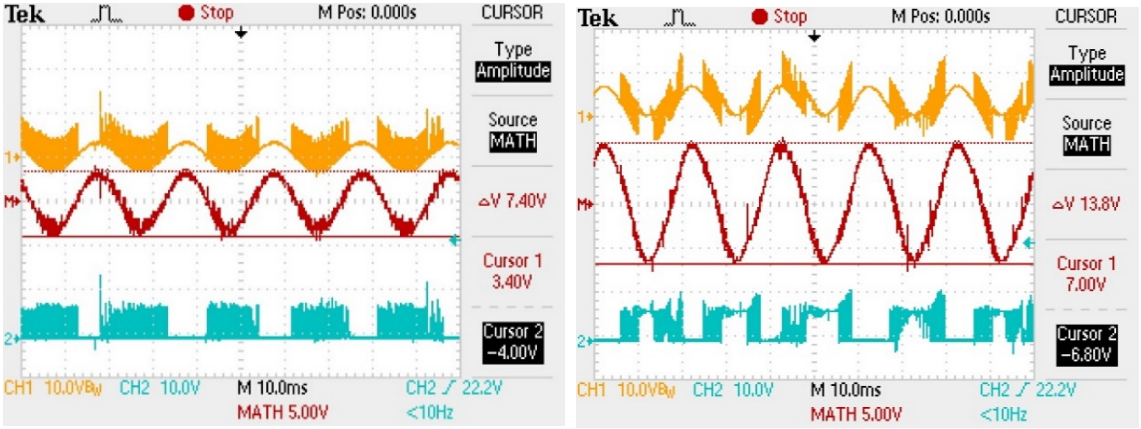


Figure 5.12: Output voltage with PWM frequency of 31.25 kHz and 62.5 kHz

However, an excessive increase of the PWM frequency will affect negatively the semiconductors, as the switching frequency is directly related to the **losses**. Thereby, it is necessary to establish the best balance between frequency and efficiency.

For the purpose of having an **estimation of the losses**, they are simulated with the software LTSpice for 31.25 and 62.50 kHz for the final schematic (Appendix D). The results are collected on Table 5.3. It is remarkable how the high-side semiconductors (1 and 3) have higher losses than the low side ones. Also, it can be verified that if the PWM frequency increases, the losses are higher as well.

	F = 31250 Hz	**F = 62500 Hz**
M1	5.678 W	11.107 W
M2	3.163 W	6.234 W
M3	5.730 W	11.099 W
M4	3.190 W	6.194 W
D1	167.760 µW	161.250 µW
D2	62.787 µW	62.179 µW
D3	167.580 µW	160.940 µW
D4	62.791 µW	62.146 µW
TOTAL	**17.763 W**	**34.635 W**

Table 5.3: Simulated losses in the MOSFETs (M) and diodes (D)

The analysis is made for a sine resolution of **8 bits**. Nevertheless, the results obtained regarding losses give reason to reconsider this resolution, as the optimal PWM frequency may be too high. As Timer 1 is selected for providing the PWM, it is possible to enhance the resolution.

As explained before, it is necessary to adequate the computation of the sine and the length of the array to the frequency selected. After testing several frequencies, it is concluded that for values **lower than 35 kHz**, the code could not compile due to memory restrictions. Between **35 and 40 kHz** the compiler awared about low free memory available but was able to deal with the code. Thus, the range of PWM frequencies that can be programmed with the code is from **35 kHz to 61.25 kHz**.

It would be expected that the higher the PWM frequency, the higher are the memory requirements. However, as explained in chapter 4, the length of the array and therefore the sample frequency are related also to the PWM frequency. The lower the frequency, the higher is the TOP value for ICR1 and as a consequence, the lengh of the array and the sampling frequency. For the current code, the memory available in the Arduino board limits the minimum PWM frequency that can be programmed.

5.2.4 V_{DC} input

Another issue to bear in mind is the relation between the input DC voltage and the output. Going again into simulation to have a first approach, different options are considered from a range between **5 and 50 Volts**. This top value is established as the voltage between drain and source in the MOSFETs selected must not be over 60V.

Also it is decided to compare the output voltage in the current design with a Half Bridge topology. The following graph collects the results obtained. The load is a resistor of 180 Ω.

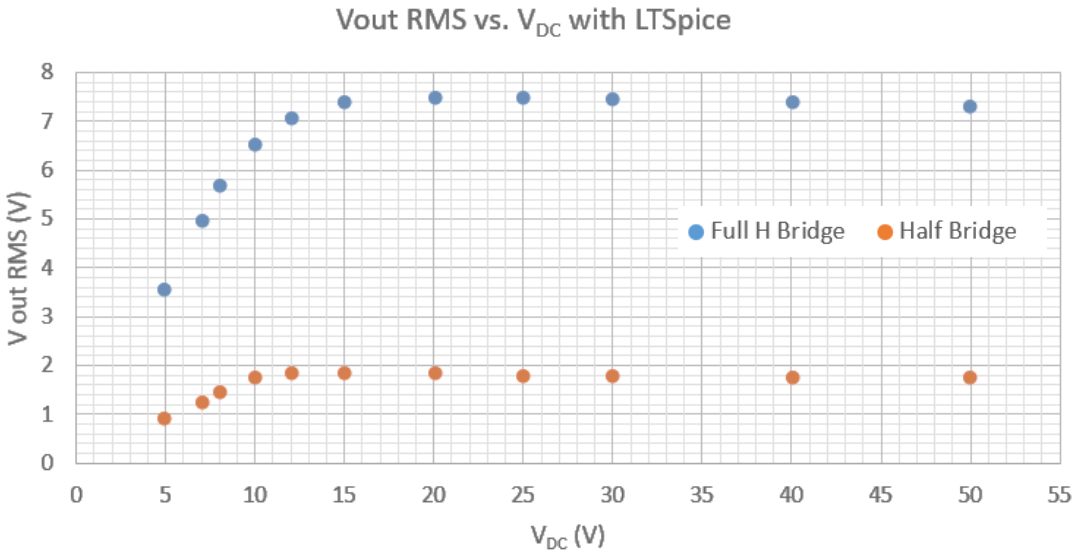


Figure 5.14: Vout (RMS) vs. V_{DC} input voltage for a Full-H Bridge and a Half Bridge.

As expected from theory, the efficiency of the Half Bridge configuration is much lower than the one of the Full H-Bridge. Also, it can be seen that the highest voltages in the load are achieved with an input of **15-25 Volts**. As the values remain similar for higher voltages, the value selected for the **DC input is 15 V**.

5.2.5 Voltages on the MOSFETs

After defining the optimal load and selecting an appropiate DC input of 15V, it is time to check if the MOSFETs are operating with the right values of voltage.

First of all, the voltage between **gate and source (V_{GS})** is tested. The Zener diode placed in the testing circuit assures that the input voltage for the gate does not boost into dangerous values, and the waveforms observed correspond to the results expected.

Figure 5.15: V_{GS} on a low-side MOSFET

Also, the **drain-to-source** (V_{DS}) voltage is checked to be oscillating between 15 Volts and 0, corresponding to the DC input. This corresponds to the simulation with LTSpice, reflected in Figure 5.16. The divisions on the vertical axis are of 2 Volts.

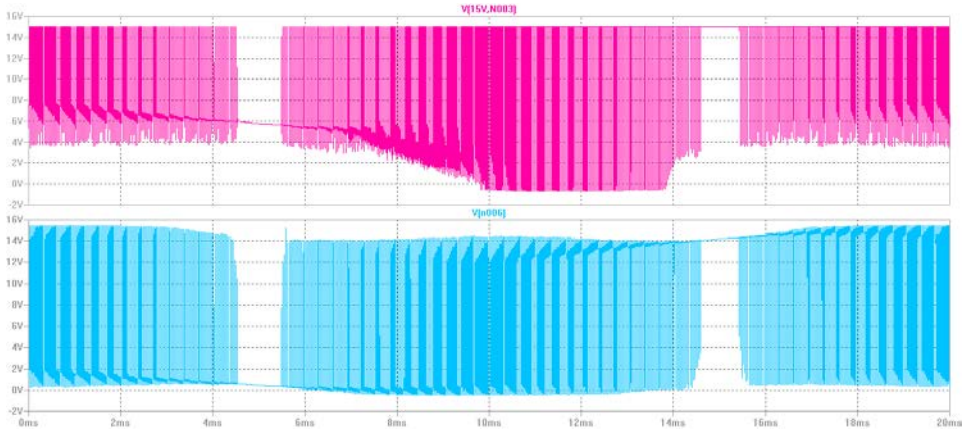
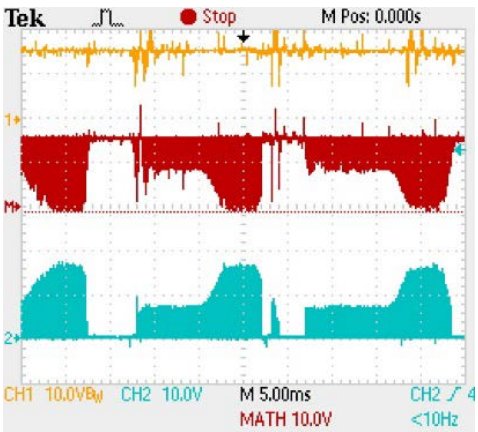
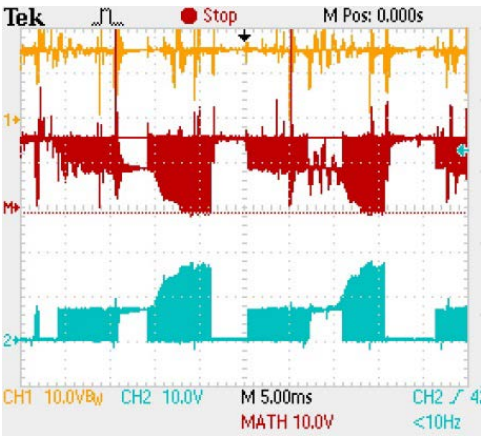


Figure 5.16: V_{DS} voltage on a high-side (top) and low-side MOSFET (bottom).

After the theoretical simulation, it is decided to check out two different PWM frequencies (31250 Hz and 62500 Hz) in the real circuit. From the results obtained in Figure 5.17, it can be seen how as the frequency is increased, the waveform has more distortion, but in both cases it oscillates between 0 and the DC input.





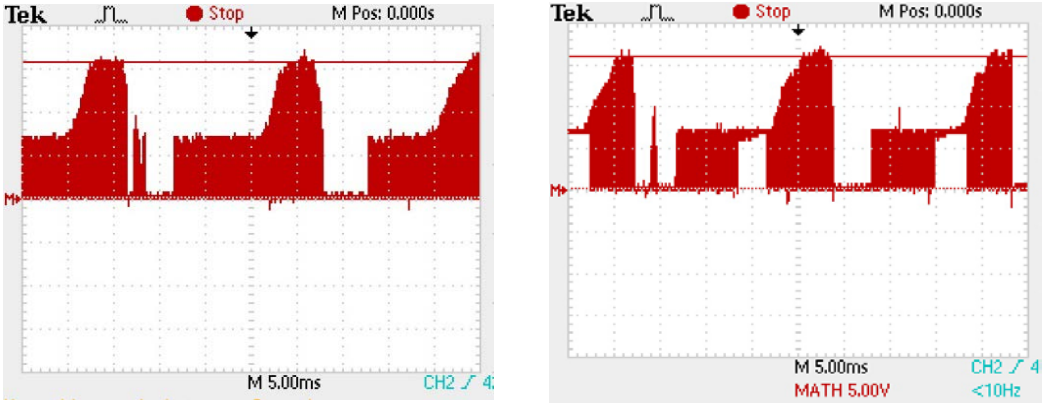


Figure 5.17: Vds on high-side (top) and low-side MOSFETS (bottom) for PWM frequency of 31250 Hz (left) and 62500 Hz (right).

5.2.6 Output voltage

Following with the analysis, the voltage on the **output without filtering** is measured and checked to be similar to the simulation. The divisions on the vertical axis on the simulation are of 4 Volts. Regarding the results, it is clear the necessity of placing a **filter** for the load.

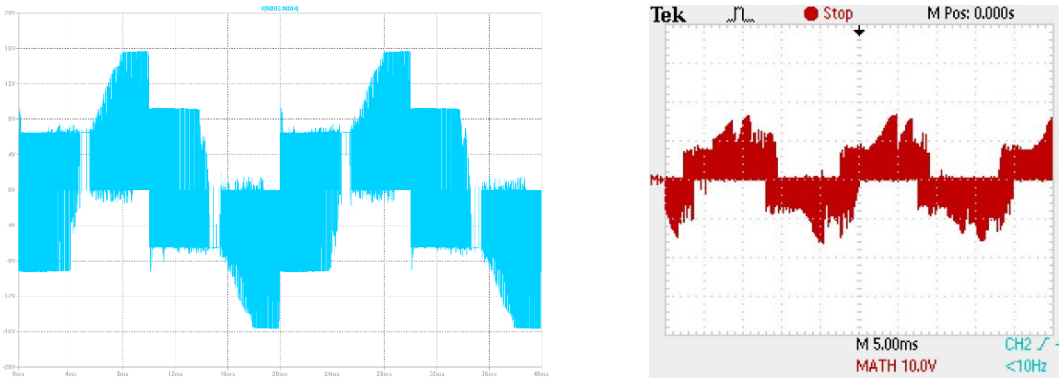
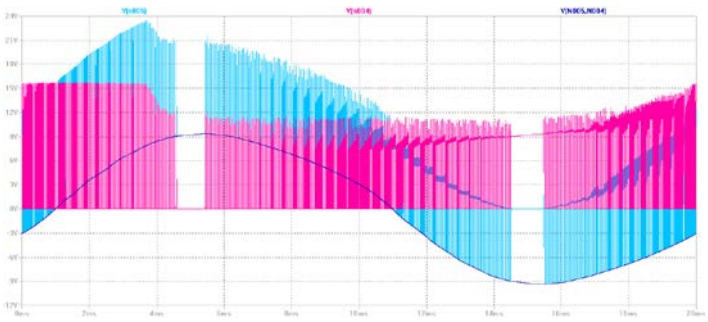


Figure 5.18: Output waveform without filter with LTSpice (left) and real (right).



The final step is to check the filtered voltage on the **load**. The simulation of the results expected is reflected in Figure 5.19.

Figure 5.19: Output waveforms. Left leg (blue), right leg (pink) and difference (dark blue) in the resistor with 31250 Hz. Vertical axis: 3V/div.

To begin with, it is analysed the variation on the **output voltage** with different **PWM frequencies** for an **8 bit resolution of the sine**. As it was analysed in Chapter 4, the value of the ICR1 register defines the TOP value of the counter for the timer and hence its resolution. By modifying this register, it is possible to obtain different PWM frequencies, but bearing in mind that there is only one that assures the maximum amplitude in the Arduino filtered output.

Thus, it is decided to examine the variation above and below the sampling frequency for 8 bits (25.5 kHz). The sine array contains 510 values, and the dutycycle varies from 0 to 255 (values for 8 bits). By leaving these values constant, it is possible to test low frequencies, as it is not needed to use high memory rates on the microcontroller.

In case of **oversampling**, for a frequency of 31.25 kHz, the RMS output is around **2.6 Volts**, value much lower than the obtained in the simulation, which is 7.41 V. This fact is due to the relation between the Arduino output and the frequency, and has been already analised. For a PWM frequency of 62.5 kHz, the RMS output goes up to **5.51 V** but the waveforms obtained are more distorted.

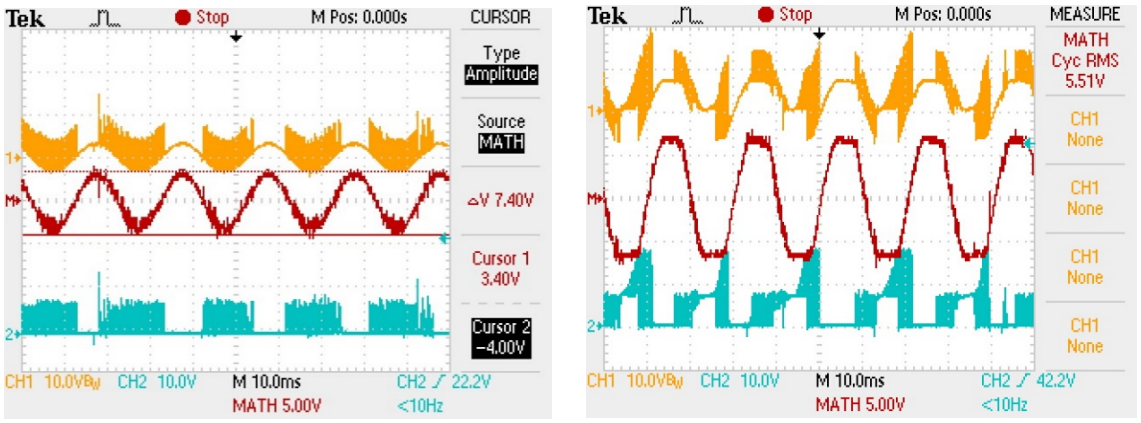


Figure 5.20: Output waveforms with 31250 Hz (left) and 62500 Hz (right)

On the other hand, in case the PWM frequency is below 25.5 kHz, the output voltage decreases considerably. For 10 kHz, an RMS voltage of 1.64V is obtained (Figure 5.21).

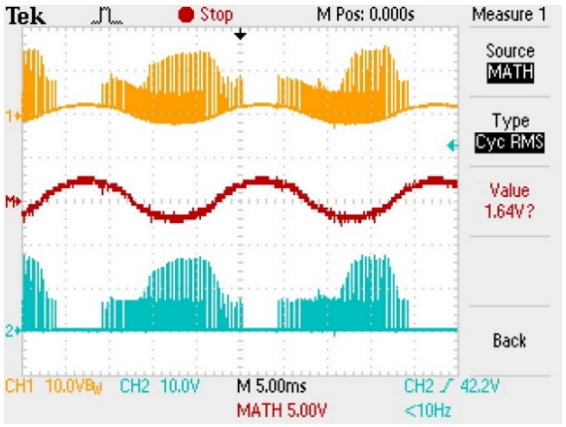
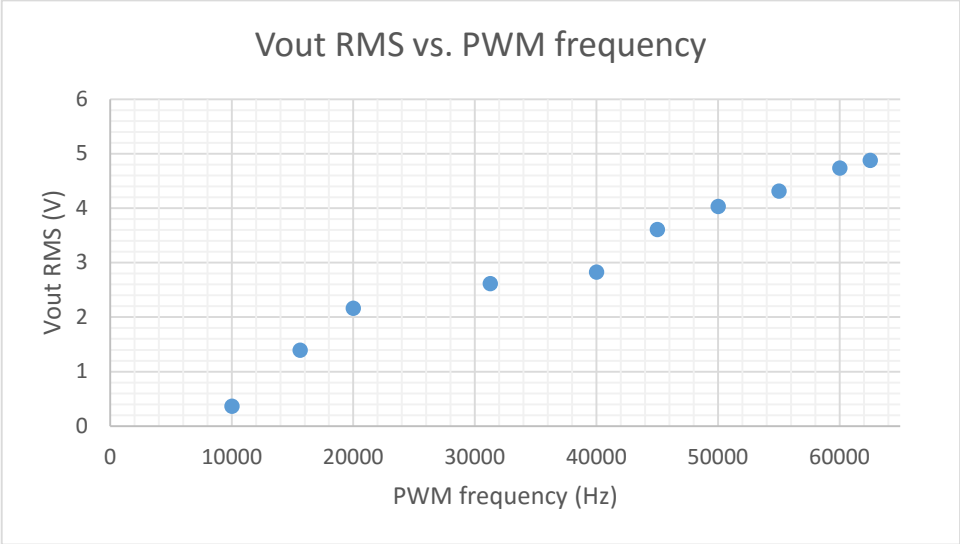


Figure 5.21: Output waveforms with 10kHz

In order to check the variation of the output regarding frequency for a sine resolution of 8 bits, a scanning for different PWM frequencies is made, being the results reflected in the graph below.



For a 15 DC input voltage, the expected values from **simulation** are around **7. 41 V**. The best results achieved in reality are around 5 Volts.

A reason for the drop in the voltage (even if the frequency is the maximum for the resolution), can be the performance of the MOSFETs. As explained previously, when a MOSFET conducts, it can be modelled as a resistor. Therefore there is a voltage drop across them proportional to the current and their internal resistance. According to their datasheet [17], the R_{DS} of the IRF60B217 is around **7.3 mΩ**. However, the current on them achieves very high spikes that can be up to the range of hundreds of Amperes (Figure 5.22, with vertical divisions of 40A). Although those spikes don't last too long in time, they are enough to make the voltage drop. On a high voltage inverter, a drop of some volts is not as crucial as in low voltage applications.

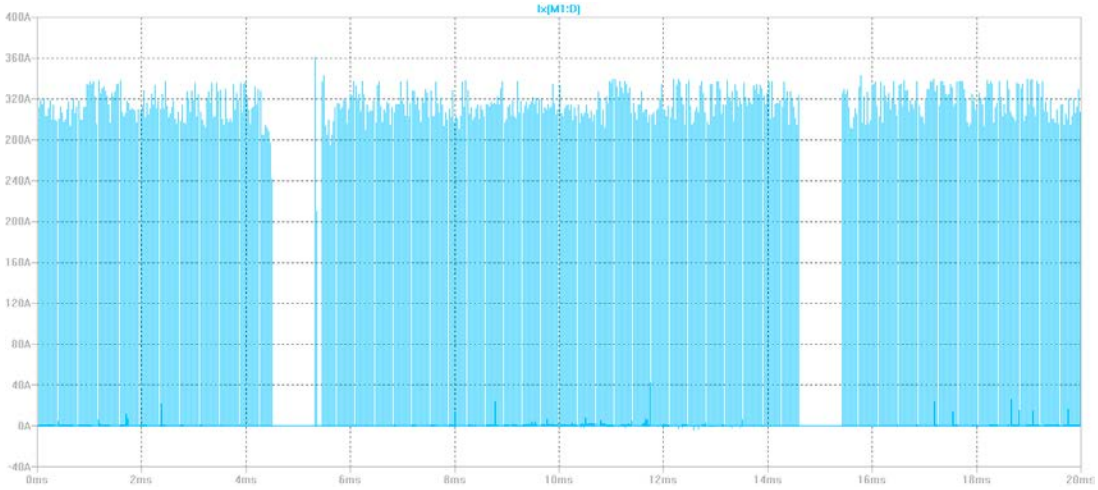


Figure 5.22: I_D on a high side MOSFET

5.2.7 Quality of the inverter

In Chapter 3 some performance criteria were set in order to define the most important aspects regarding the results obtained. In this section they will be reviewed with application to the current design.

- **Efficiency**

To determine the efficiency of the complete circuit, it is necessary to know the input and the output power. As an example, **15 V** are set as DC input voltage and some computations will be made.

The DC input power supply allows the display of the current provided to the circuit, and it is verified that in any case this value exceeds **0.1 A**. Thus, this will be the value selected.

Regarding output power, it is possible to relate the RMS voltage obtained to the value of the load. For the circuit tested, the load is **180 Ω.** The maximum voltage obtained is **5.51 RMS V**, for a 62.5 kHz PWM frequency and a resolution of 8 bits.

With these values, it is possible to determine the efficiency of the circuit as:

$$\eta = \frac{P_{out}}{P_{in}} . 100\ (\%)$$

$$P_{in} = V * I = 15 * 0.1 = 1.5\ W \qquad P_{out} = \frac{V^2}{R} = \frac{5.51^2}{180} = 0.1687\ W$$

$$\eta = \frac{0.1687}{1.5} . 100\ (\%) = \mathbf{11.25}\ \%$$

The theoretical efficiency for this results with an output voltage of 7.41 V is

$$\eta = \frac{\frac{7.41^2}{180}}{1.5} . 100\ (\%) = \mathbf{20.33}\ \%$$

- **Quality of the waveform**

The second issue to consider is the quality of the waveform. As it is observed in the oscilloscope, the signal obtained has some imperfections. This is directly related to the harmonic distortion of the sine.

Determining the Total Harmonic Distortion (THD) of a signal is possible by transforming the time domain signal into frequency domain. The most common way to do that is with a **spectrum**

analyser. This equipment shows the amplitude of the signals on a logarithmic scale (dB) with the aim to convert them into percentage of THD.

Another procedure is to compute the Fast Fourier Transform, which is possible with the *Math* function of the **oscilloscope.** Relating the amplitude of each peak to the fundamental and thereby knowing the values of the voltage for the harmonics, the THD can be computed as explained in section 3.1.2 of Chapter 3.

Due to the possibilities regarding material available, this second option was considered as a way of measure the THD of the output signal. However, it was soon confirmed that it was not possible to apply it, as Math functions must be related to the measurements on the channels of the scope. As the output signal itself is obtained by subtracting the voltage on the two legs of the load with the Math function, it is not possible to apply this tool again for the THD.

Thus, in case the THD is desired to be measured, it should be done with the spectrum analyser or with a Picoscope. Applying the formula in chapter 3, it is possible to obtain the percentage of harmonic distortion. Some recommendations regarding decreasing the THD are the placement of the elements as closed as possible. As the circuit will be printed on a PCB, this fact will help to reduce harmonics.

- **Reactive power**

The output power obtained in the circuit depends on the kind of load placed as output. Considering the load after filtering, in the circuit tested it is a simple resistor and therefore current and voltage are in phase and the energy flow goes only in one direction. As a result, only **active power** is transmitted between the source and the load, and the **power factor is 1**.In case the reactive power is desired to be tested, a capacitive or inductive load should be placed.

However, in case the inductance of the filter is considered, it introduces a phase difference between current and voltage and the power factor decreases.

In higher voltages, motors or transformers are examples of inductive loads. By measuring the phase difference between current and voltage it could be possible to determine the power factor of each specific case.

5.3 PCB

After testing on the protoboard, the next step is the implementation of the circuit on a **PCB**. Printed Circuit Boards allow the connection of electronic components thanks to the conductive tracks printed on cooper sheets laminated on a non conductive substrate. It is possible to have more than one layer of copper on them, which allows to build complex circuits. The elements of the circuit are soldered to the copper.

The software employed for the PCB design is EAGLE. With several libraries and wide information and support on the internet, it is a common tool for PCB layout design. It is useful to check the design rules of PCB manufactures to avoid mistakes not only while designing, but also at the time of manufacturing.

The procedure for designing is common to almost every circuit. First, the design is drawn in an schematic. The concrete elements have to be found in libraries with the right packages. The software allows modifying the elements or creating new symbols and packages.

Once the schematic corresponds to the circuit desired, the elements are placed on the board. The big elements have to be firstly allocated (the drivers and the inverter gate in this case). The position in the board is also important. The MOSFETs and the voltage regulator should be placed near the edge so they can be mounted on a heat sink if necessary.

The drill distances, as well as the dimension of the elements have to be carefully checked. Also the drill size is important, as an example, the inductance drills in the current design are bigger than the resistors. For easier mounting the PCB on a plate, the mounting drills and outlines should be in a metrical grid (by coordinates).

In order to draw the paths, the software contains an Autorouter tool. However, the best option is to draw them by hand, as usually it employs two layers or the paths are not wide enough. In the current design only one layer is required. Thus, the cost of the design is lower, but the connection of the elements is not as easy as with two layers.

The current prototype will be manufactured with a moulding cutter that belongs to the university. The cutter can only properly handle horizontal, vertical and 45º angles tracks. As a general rule, perpendicular junctions should be avoided, and the junctions should be made in the pins. Also there should be a contiguous area (polygon) connected to ground, to isolate adjacent connections.

The complete schematic of the circuit, as well as the board layout are collected in Appendix D.

Chapter 6

Conclusion

In this report, a complete description of the procedure followed for designing a single-phase inverter has been developed. As a resume, the following points have been described:

- **Theoretical approach** to the single-phase inverter. Definition, applications and topologies, as well as an explanation of its elements.

- Definition of the **design criteria** and **selection of the specific components**. Discussion of the best options regarding modulation, switching frequency and topology.

- Justification of the **elements placed** in the design and their value.

- Analysis of the **microcontroller**. Possibilities regarding PWM generation.

- **Testing** of the circuit on a protoboard. Discussion of the results obtained and comparison with simulating expectations.

6.1 Objectives achieved

The main purpose of obtaining a pure sinusoidal waveform with digital implementation is met. The PWM generation is achieved with a simple C code, as a convenient substitute of analog circuitry. Thus, in case some changes are desired to be made regarding signal or PWM frequency, it is only necessary to change the corresponding values of the registers.

Also, as the circuit is built using discrete components instead of an integrated circuit, it is easier to check the performance of the elements that constitute the bridge.

However, the levels of voltage achieved in the output are a little lower than the ones expected from simulation. One possible reason for this is the performance of the real elements. Voltage drops and power consumption are some effects that cannot be easily predicted and that have high influence on the results obtained.

Regarding the C code, to have the optimal output in the Arduino, the PWM frequency is limited on a range from 35 to 62.25 kHz. This fact is due to the memory available in the microcontroller selected.

6.2 Further developments

Some recommendations and guidelines are also collected with regard to future improvements of the circuit. The lack of time is the main reason of their absence in the current work.

To determine the quality of the inverter, the best option would be applying the spectrum analyser or the Picoscope to have an idea of the distortion in the waveform. Also, to analyse the variation of power factor, different inductive or capacitive loads could be placed as output. 7

6.2.1 Efficiency

The efficiency tested on the circuit is around **11%**. However, it would be necessary to check the values of the current more carefully to have a better idea of the exact figure. Taking this estimation, the first idea that comes into mind to increase this value. As explained in previous pages, switching frequency on the semiconductors is directly related to this fact, and therefore it should be kept on values around 30-40 kHz on the current design. Also there are some considerations that can be taken into account to increase efficiency, as selecting another power supply or trying with other PWM modulations.

- **PWM modulation**

Although unipolar has been the most widespread solution regarding PWM modulation, the current techniques in power electronics are mostly focused on **modified unipolar modulation** as an option to increase efficiency (e.g. [13]). In this case, the high-side transistors switch at high frequency (in the order of kilohertz) and the low-side at grid frequency, depending on the sign of the control signal. Whereas the switching losses dominate the high-side transistors, they are almost insignificant in the low-side devices. As a whole, the efficiency achieved is similar or even better than in unipolar modulation.

Thus, as an idea of further guideline, the digital implementation of this modulation can be also considered. Due to the lack of analog circuitry related to the PWM generation, fortunately only the code needs to be adapted.

Also, it is possible to select different models of transistors for the high-side and low-side switchers. Thereby, the MOSFETs are optimized for their correspondent switching frequency and have lower losses.

6.2.2 Microcontroller

For the current project the microcontroller selected is Arduino UNO. As explained before, the optimal output signal of the board with the current C code is obtained in a range of PWM frequencies from 35 kHz to 62.5 kHz. As the top values of the dutycycle and the length of the sine array are related to the PWM frequency, the compiler warned of memory limitations for the lowest values of frequency in this range.

In case the resolution is desired to be improved, the output obtained in the Arduino is not the optimal. Thus, it would be necessary to consider other options of microcontrollers to achieve lower PWM frequency values.

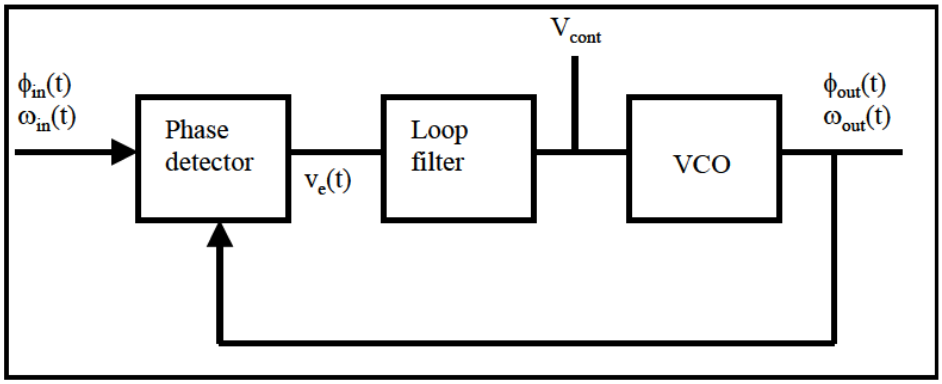
Also, with regard to a future control implemented, it is possible to select a more powerful option, as Arduino Due or Nano.

6.2.3 Control

Despite the implementation of a control system was considered at first, it is left to further work due to limitation in time.

Depending on the application of the circuit, the control applied varies. In this case, the option is to make it a grid connected inverter. Thus, a **phase loop** is considered as the option to implement.

PLL (Phase – Locked Loops) are control systems that generate an output signal whose phase is related to the phase of an input signal. Their main blocks are the phase detector, a low pass filter and a Voltage Controlled Oscillator (VCO).



Normally this system is developed with analog circuitry. However, as the PWM has already been generated digitally, it is only necessary to modify the code in order to add the control.

Some issues should be taken into account, as how to implement the **phase detection** and specially the **voltage oscillator**. The phase difference measured between the reference and the input has to be related to the value of the **dutycycle**, and therefore included on the Interrupt Service Routine of the current code.

References

[1] MARROYO, LUIS and MARCOS, JAVIER. *Convertidores Electrónicos de Potencia*. Universidad Pública de Navarra, 2015

[2] *AC vs. DC*. Differ; Homepage, Science, Physics. http://www.diffen.com/difference/Alternating_Current_vs_Direct_Current

[3] *Will the World's Emerging Economies Shape the Future of the Electric Grid?* The Energy Collective. http://www.theenergycollective.com/stevencollier/285686/will-worlds-emerging-economies-shape-future-electric-grid

Battery Charging On Board Ship. Marine Insight. http://www.marineinsight.com/marine-electrical/battery-charging-on-board-ship/

Wireless Control of AC Devices with a Power Switch Tail. http://examples.digi.com/lights-motors-more/wireless-control-of-ac-devices-with-a-powerswitch-tail/

DC-AC POWER-INVERTER Series. DC/AC Energie Philippines. http://dc-ac-phil.weebly.com/

[4] J. DOUCET, D. EGGLESTON, J. SHAW. *DC/AC Pure sine wave inverter*. Worcester Polytechnic Institute.

[5] *Application Manual Power Semiconductors*, SEMIKRON. 2nd Edition, available in: https://www.semikron.com/dl/service-support/downloads/download/semikron-application-manual-power-semiconductors-english-en-2015

[6] PARK, JONG-MUN. *Evolution of power semiconductor devices.* 2004-10-28 http://www.iue.tuwien.ac.at/phd/park/node14.html

[7] *Power electronics*, Wikipedia. https://en.wikipedia.org/wiki/Power_electronics

[8] BARNETXEA, AITOR. *Sistema fotovoltaico aislado: inversor monofásico*. Bachelor Final Thesis, 2012. Universidad Pública de Navarra.

[9] ESPARZA, TOMÁS. *Diseño, dimensionado y simulación de un convertidor DCDC.* Trabajo Fin de Grado. Universidad Pública de Navarra.

[10] MARTÍN, RAÚL. *Diseño e implementación experimental de un inversor monofásico operando en modo isla*. Trabajo Fin de Grado. Universidad Carlos III.

[11] *Total Harmonic Distortion and Effects in Electrical Power Systems,* Associated Power Technologies http://www.aptsources.com/resources/pdf/Total%20Harmonic%20Distortion.pdf

[12] *IGBT vs. MOSFET: Which Device to Select?* Renesas Electronics America Inc, 2012.

[13] *Build an Efficient 500-W Solar-Power Inverter Using IGBTs*. WIBAWA T. CHOU. Electronic Design; Home; Markets; Energy http://electronicdesign.com/energy/build-efficient-500-w-solar-power-inverter-using-igbts

[14] *Thermal design* http://www.smps.us/thermal.html

[15] Types of capacitors, Learning about electronics. Articles; Types of Capacitors. http://www.learningaboutelectronics.com/Articles/Types-of-capacitors

[16] Arduino UNO. https://www.arduino.cc/

[17] *IRF60B217 Datasheet*, International Rectifiers. Home; Power MOSFETs; 20V-300V N-Channel Power MOSFET; 40V-75V N-Channel Power MOSFET ; IRF60B217; Datasheet

http://www.infineon.com/dgdl/irf60b217.pdf?fileId=5546d462533600a4015355e42b3719c8

[18] *IR2110 Datasheet*, International Rectifiers. Products; Power; Motor Control & Gate Driver ICs; Non-Isolated Gate Driver IC; General Purpose Gate Driver ICs-Industrial; IR2110, available in:

http://www.infineon.com/dgdl/ir2110.pdf?fileId=5546d462533600a4015355c80333167e

[19] *Application Note AN-978, HV Floating MOS-Gate Driver ICs*. International Rectifier-Infineon.

http://www.infineon.com/dgdl/an-978.pdf?fileId=5546d462533600a40153559f7cf21200

[20] *Snubber design for noise reduction in switching circuits*. AN100, Alpha&Omega semiconductors. S. Havanur. Alpha&Omega semiconductors; Homepage; Application notes; AN 100. http://www.aosmd.com/res/application_notes/mosfets/AN100_Snubber_Design_for_Noise_Reduction.pdf

[21] *Atmel, Atmel 8-bit microcontroller with 4/8/16/32 Kbytes in-system programmable flash datasheet.* Home; Products; Microcontrollers; AVR 8- and 32-bit MCUs; Atmega 328.

http://www.atmel.com/images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf

[22] *LM7805C voltage regulator Datasheet*, Texas Instruments. TI Home; Semiconductors; Power Management; Linear Regulator (LDO); Single Channel LDO; LM7805C 5 Volt Regulator, available in: http://www.ti.com/lit/ds/symlink/lm7805c.pdf

[23] Secrets of ARDUINO PWM. Ken Shirrif's blog.

http://www.righto.com/2009/07/secrets-of-arduino-pwm.html

[24] *SN74L04, Hex Inverters Datasheet*. Texas Instruments.

http://www.ti.com/lit/ds/symlink/sn74ls04.pdf

[25] *How To Decouple A Logic Circuit?* SLEIGHT, KENNETH. 2010. Bright Hub Engineering. Home; Electrical Engineering; DIY Digital/Analog Electronics

http://www.brighthubengineering.com/diy-electronics-devices/56501-how-to-decouple-a-logic-circuit/?PageSpeed=noscript

[26] *Choosing and Using Bypass Capacitors, Application Note 1325*. Intersil. http://www.intersil.com/content/dam/Intersil/documents/an13/an1325.pdf

[27] *Power MOSFET*, Wikipedia

https://en.wikipedia.org/wiki/Power_MOSFET

[28] *AN-6076 Design and Application Guide of Bootstrap Circuit for High Voltage Gate Drive IC*. Fairchild

https://www.fairchildsemi.com/application-notes/AN/AN-6076.pdf

[29] *What is the function of the series gate resistors in the gate drives of IGBTs and HEXFETS?* Infineon Custom Support.

http://irf.custhelp.com/app/answers/detail/a_id/215

[30] *Power MOSFET Basics*. International Rectifier.

http://www.infineon.com/dgdl/mosfet.pdf?fileId=5546d462533600a4015357444e913f4f

[31] *Prescaler*. Wikipedia.

https://en.wikipedia.org/wiki/Prescaler

[32] *Arduino Interrupts*, EngBlaze

http://www.engblaze.com/we-interrupt-this-program-to-bring-you-a-tutorial-on-arduino-interrupts/

[33] *PCB design guidelines*. Eurocircuits.

http://www.eurocircuits.com/PCB-design-guidelines

Appendix A: Calculations

On the following pages, a further explanation of some of the computations made to determine the elements of the circuit is developed. In some cases the origin of the formula is referenced to the source applied, but in the case of the output inductance, the formula is deduced.

A.1 Bus DC capacitor

Placed after the voltage source, its aim is to damp the voltage ripple resulting from a pulsating current that appears on the DC side of the inverter. The frequency of this current is twice the output frequency. In [1] the value of the peak to peak ripple voltage is determined to be:

$$\Delta V_{DCpp} = \frac{V_{AC}\, I_{AC}}{C\ \omega\ V_{DC}}$$

From this expression, the value of the DC capacitor can be computed as:

$$C \geq \frac{P_{nom}}{\Delta V_{DC}\ .\ \omega\ .\ V_{DC}}$$

Where

- Pnom is the nominal power, measured in Watts.
- ΔV_{DC} is the maximum ripple voltage desired in the capacitor.
- ω is the output frequency of the inverter.
- V_{DC} is the voltage in the DC side.

The maximum ripple voltage is desired to be 10%. The input DC voltage is 15 volts and the desired frequency is 50 Hz. Regarding power, the inverter would be desired to deal with around 25-30W. For security reasons, the nominal power of the inverter will be taken as 30 W. Taking all of this into account, the minimum capacitance that can be used in the circuit is:

$$C \geq \frac{P_{nom}}{\Delta V_{DC}\ .\ \omega\ .\ V_{DC}} = \frac{30}{0{,}1\ .\ 15\ .\ 2\pi 50\ .\ 15} = \mathbf{4.25\ mF}$$

It may be shocking the relatively high value of the capacitance. However, it is necessary to take into account that the values for input voltage and output frequency and very low. Also, this value is indicative and will be the highest achieved, as the power figure is set as limit. For this reason, a capacitance of **22µF** is enough for the design.

A.2 Output inductance

The determination of the value for the output inductance depends on the PWM modulation selected. As unipolar modulation is chosen in the current project, the formulas will be deducted for this specific case. Although PWM is generated digitally, the procedure followed to determine the inductance will be the same as in analogical implementation.

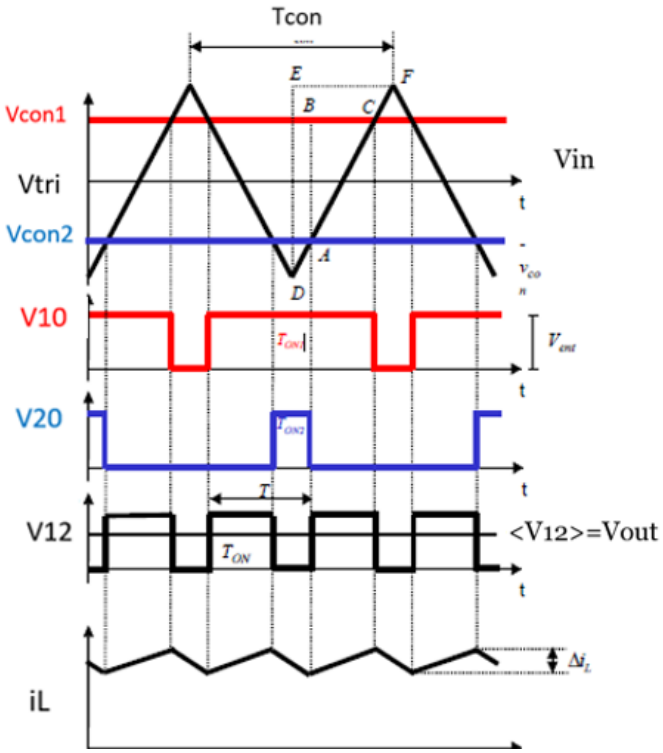


Figure A.1: Unipolar PWM waveforms [1]

Figure A.1 shows the main waveforms of analogical unipolar PWM modulation. The output voltage (V12) is obtained by the difference between the voltages in both branches (V10 and V20).

As a resume of the explanation in section 2.8.2, the triangular voltage (Vtri) defines the switching period (Tcon) and for each branch the correspondent voltages (V10 and V20) are obtained with comparators that relate the control voltage (Vcon) with Vtri. The outputs of these comparators are the connection functions of the MOSFETs.

The output voltage is computed as

$$V_{12} = V_{10} - V_{20}$$

As it can be seen graphically in Figure A.1, the period of V12 (T) is half the switching period (Tcon). This output period must not be misled with the 50Hz output sine, as it corresponds to the waveform before the filter. To clarify the procedure followed, a basic schematic of an H-Bridge is shown in Figure A.2.

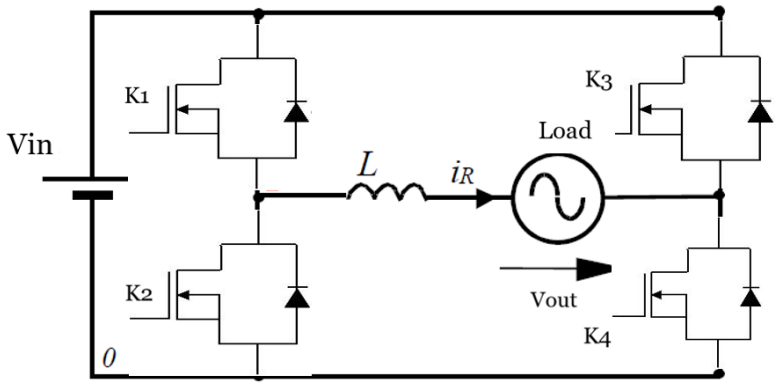


Figure A.2: H-Bridge schematic

The voltage on an inductance can be computed as:

$$V_L = L\,\frac{di_L}{dt}$$

During Ton (semiconductors K1 and K4 conducting on figure A.2), this voltage is

$$V_L = V_{in} - V_{out} \rightarrow (V_{in} - V_{out})\,dt = L\,di_L$$

As the sine output frequency is much lower than the switching frequency, the output voltage can be considered as constant for computations:

$$\int_0^{T_{ON}} di_L = \Delta i_L = \frac{1}{L}\int_0^{T_{ON}} (V_{in} - V_{out})dt = \frac{(V_{in}-V_{out})\,T_{ON}}{L} \qquad (1)$$

From the graph in A.1, it can also be deduced the relationship between PWM generation voltages and the duty cycle.

$$\frac{B-C}{E-F} = \frac{T_{ON}}{T} = \frac{A-B}{D-E} = \frac{2V_{con}}{2V_{pt}} = m$$

m is defined as the modulation index, and relates the control voltage and the triangular wave on the PWM.

Considering this, the average output voltage can be expressed employing the modulating as follows:

$$< V_{12} >= V_{12f} = \frac{T_{on}}{T}\,V_{in} = m\,.\,V_{in}$$

Also, the output voltage (Vout) can be approximated to the fundamental component of V12 ($< V_{12} >$, the one at 50 Hz). Applying this to expression (1), it can be seen that ripple depends on the modulation index.

$$V_{out} =< V_{12} >= V_{12f}$$

$$\Delta i_L = \frac{(V_{in} - V_{out})\, T_{ON}}{L} = \frac{(V_{in} - m\, V_{in})\, m\, T}{L} = \frac{V_{in}\,(1 - m\,)\, m}{2\, L\, F_{switch}}$$

In order to include the switching frequency (F_{switch}) in the formula, it is necessary to multiply by 2, as the output frequency is twice the switching frequency.

To establish the maximum rippling current (Δi_{Lmax}) the function must be maximized.

$$\frac{\partial \Delta i_L}{\partial m} = 0 \rightarrow m = \frac{1}{2} \rightarrow \Delta i_{Lmax} = \frac{V_{in}}{8\, L\, F_{switch}}$$

The value of the maximum rippling current desired should be defined in each case. After that, the minimum inductance that assures this value can be determined as:

$$L \geq \frac{V_{in}}{8\, \Delta i_{Lmax}\, F_{switch}} \qquad (2)$$

In the actual design, the value of the current in the inductance can be computed taking as data power and voltage, and assuming current and voltage are in phase ($\cos\varphi = 1$).

$$P = V.I\,.\cos\varphi \rightarrow I = \frac{P}{V} \qquad (3)$$

To have an estimation, the power will be set into 35 W and the output voltage is 10.6 effective volts. From (3) an output alternating current of 3 A effective as top value can be computed. A maximum rippling of 5% is desired.

Substituting variables in expression (2):

- Input voltage, Vin= 15 V.
- Switching frequency, F_{switch}= 31250 Hz
- $\Delta i_{Lmax} = 5\%\, i_L = 0.167\, A$

$$L \geq \frac{V_{in}}{8\, \Delta i_{Lmax}\, F_{switch}} = \frac{15}{8\,.\,0{,}167\,.\,31250} = \mathbf{360\ uH}$$

As the frequency can vary on the design and the current value is not very high, in order to have lower oscillations in the current an inductance of **470µF** will be selected.

A.3 Filter

In order to design the LC filter, it is necessary to determine the **cut-off frequency** (fc) above which frequencies are attenuated.

$$wc = \frac{1}{\sqrt{LC}}\left(\frac{rad}{s}\right) \rightarrow fc = \frac{1}{2\pi\sqrt{LC}}\ (Hz)$$

Having chosen unipolar modulation, the output frequency is twice the switching frequency, which is 31.25 kHz. Thus, the **output frequency** will be 62.50 kHz. As the inductance is the same employed to smooth the output current, its value is already fixed on **470 µH.**

The cut-off frequency also depends on the value of the capacitor of the filter. This way, one of the variables (cut frequency or capacitor) must be fixed in order to obtain the other. As the values of the capacitors are standard, the procedure chosen will be that of selecting a capacitor which assures a satisfactory output waveform. This way, on a rough range between 50 and 500µF, it will be chosen a capacitance of **47µF**. Thereby, the cut-off frequency will be

$$fc = \frac{1}{2\pi\sqrt{470\,.\,10^{-6}.47.10^{-6}}} = \mathbf{1070,83\ Hz}$$

This value is high enough not to filter the output voltage at 50 Hz but much lower than the switching frequency. Therefore only the frequencies desired are obtained.

In conclusion, a low pass **LC filter** with a series inductance of **470 µH** and a parallel capacitance of **47 µF** is selected.

A.4 Gate to source resistor

The aim of the gate to source resistor is to discharge the gate capacitance C_{GS} in case there are parasitic charges due to leakage currents. A simple procedure can be followed in order to determine its maximum value.

The voltage in a capacitance can be computed as

$$V_c = \frac{1}{C}\int i_c(t)dt$$

When the IRF60B217 is turned off, the gate-to-source leakage current is 100 nA (from its datasheet).

The minimum gate voltage is set in 2.1 volts. As the gate capacitance (C_{GS}) is not given as data, the reverse transfer capacitance (C_{rss}) is substracted from the input capacitance (C_{oss}).

$$C_{rss} = C_{GD} = 140\,pF \longrightarrow C_{GS} = C_{iss} - C_{rss} = 2090\,pF$$

$$C_{iss} = C_{GD} + C_{GS} = 2230\,pF$$

Therefore, the time necessary for charging the gate capacitance by the leakage current is:

$$t_{min} = \frac{V_c \,.\, C_{GS}}{I_{GS}} = \frac{2{,}1 \,.\, 2090 \,.\, 10^{-12}}{100 \,.\, 10^{-9}} = 4.3890\,ms$$

τ_{charge} can be defined as the time needed in order the gate capacitance to achieve the 63,2 % of the total charge.

$$\tau_{charge} = 0{,}632 * \ 4{,}389.10^{-3} = 2.7738\,ms$$

In an RC circuit, the time constant τ can be computed as $\tau = RC$. The time constant of discharge must be lower than the parasitic charge, in order to avoid it and to protect the MOSFETs. Thus, the maximum value that R_{GS} can have is

$$R_{GS} < \frac{\tau_{charge}}{C_{GS}} = \frac{2{,}7738 \,.\, 10^{-3}}{2090 \,.\, 10^{-12}} = \mathbf{1{,}327\ M\Omega}$$

The value selected for the resistor will be a standard of **47 kΩ.**

A.5 Snubber circuit

As explained in the Application Note AN100-1 form Alpha-Omega [20], in order to determine the values of the snubber resistor and capacitor, it is necessary to have a previous estimation of the parasitic inductances. These are undesirable effects in the circuit, and their sources are the traces between input capacitor and the switching devices. In practical circuits, they cannot be eliminated.

The optimum value of the snubber resistor must equal the characteristic impedance of the LC circuit [20].

The output capacitor of the low-side MOSFET (Coss) is the main source of parasitic capacitances that may ring with these inductances. For the IRF60B217 this value corresponds to 215 pF with a drain-to-source voltage of 25 volts. This value can vary with lower supply voltages, but having a glimpse into Fig. 7 of the datasheet, for $V_{DS} = 10\,V$ can also be taken as valid.

To estimate the value of the parasitic inductances, as a practical method it is possible to measure the frequency in the ringing waveform. Figure A.3 reflects the results obtained for the gate-to-source voltage. As it can be seen, the voltage oscillates during two periods in a range of 4 volts. Measuring the time with the cursors, it is possible to determine the period of one pulse (T_{ring}).

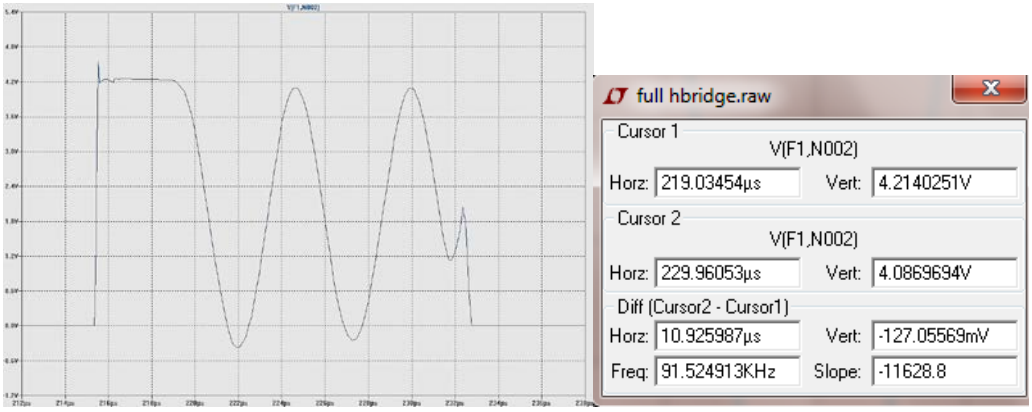


Figure A.3: Ringing drain-to-source voltage.

$$T_{ring} = \frac{time\ measured}{number\ of\ periods} = \frac{10{,}9259\ .10^{-6}}{2} = 5{,}4629\ \mu s$$

Following the calculation procedure of [20], the total estimated parasitic inductance (L_{ckt}) can be computed as

$$T_{ring} = 2\ \pi \sqrt{L_{ckt} C_{oss}}$$

$$L_{ckt} = \frac{{T_{ring}}^2}{4\ \pi^2\ C_{oss}} = 3.5160\ mH$$

The snubber resistor needed for damping and also for avoiding the instantaneous discharge of C_{snub} is

$$R_{snub} = \sqrt{\frac{L_{ckt}}{C_{oss}}} = \sqrt{\frac{3{,}516.10^{-3}}{215.10^{-12}}} = 4043.97\ \Omega \approx \mathbf{4}\ \boldsymbol{k}\Omega$$

In order to determine the value for the snubber capacitor, it is necessary to take into account that large values provide over damping and reduce the number of oscillations. This capacitor also stores and dissipates energy every cycle. As the energy dissipated is not very high, it would not be very harmful. It is usual to multiply the ringing period by a constant to compute the time constant of the RC circuit.

The $R_{snub}C_{snub}$ time constant can be expressed as a multiple of the ringing period [20]. Usually 3 is the minimum value selected.

$$C_{snub} = \frac{3 \,.\, T_{ring}}{R_{snub}} \approx \mathbf{4\ nF}$$

To provide extra damping, a value of **10 nF** will be selected. The energy stored in the capacitor can be computed as

$$E_{snub} = \frac{1}{2}\, C_{snub} . {V_{in}}^2 = \frac{1}{2}\, 4.10^{-9}.15^2 = 450\, nJ$$

Taking into account the switching frequency (F_{sw}) he power in the capacitance is

$$P_{snub} = E_{snub} \,.\, F_{sw} = 450\,.10^{-9}\,.31250 = 14.06\, mW$$

The snubber resistor should dissipate this power. The value selected for it is **500Ω.**

A.6 Heatsink

In order to determine the necessity of placing a sink or not in the circuit, some simple calculations can be made. As briefly explained in Chapter 3, the losses that the elements can dissipate depend on their thermal resistors and the maximum temperatures.

$$P_{lost} = \frac{T_{jmax} - T_{amax}}{(R_{thjc} + R_{thch} + R_{thha})}$$

From the datasheet of the MOSFET IRF60B217, it is possible to establish the maximum temperature for the junction, which is said to be 175℃. Also, the maximum ambient temperature will be fixed in 40℃.

Regarding the thermal resistors, from the datasheet there are some values available (Figure A.4).

Thermal Resistance

Symbol	Parameter	Typ.	Max.	Units
$R_{\theta JC}$	Junction-to-Case ⑦	—	1.8	°C/W
$R_{\theta CS}$	Case-to-Sink, Flat Greased Surface	0.50	—	
$R_{\theta JA}$	Junction-to-Ambient	—	62	

Figure A.4: Thermal resistances of IRF60B217 [Appendix B]

Thus, in case a sink is not placed, R_{thja} is 62 ℃/W . The maximum dissipated losses are

$$P_{lost} = \frac{175 - 40}{62} = \mathbf{2,1774\ W}$$

This value is pretty low for the current application, according to the estimation made with LTSpice. Therefore, a **sink will be attached to every MOSFET.**

Appendix B: Code

In this section, the complete code for the Arduino board is developed, as a complement for the explanation given in previous pages. Also, a table with some PWM frequencies and the values for the registers according computations can be found. It is recommended to write the values by hand instead of relating them to a formula. Otherwise the program does not recognize them and the compiler does not work.

f_pwm	ICR=TOP	n	TOP/2		OCR	
				fs (Hz)	prescaler 8	prescaler 1
62500	255	510	127.5	25500	77	626
60000	266	532	133	26567	74	601
55000	290	581	145	28991	68	551
50000	319	638	159.5	31900	62	501
45000	355	710	177.5	35456	55	450
40000	399	798	199.5	39900	49	400
35000	456	912	228	45614	43	350

Table B1: Values computed for different PWM frequencies.

```
1   //.....VARIABLES DEFINITION.....
2    // int f_signal=50;              // Hz. Frequency desired for the output sine.
3   int f_pwm=62500;                 // Hz. PWM frequency.
4   // int TOP (int) (16000000/f_pwm)-1;        // Top value for the resolution selected.
5   int TOP= 255;
6   int n= (TOP * 2);                // Length of the array.
7   int n= 510;
8   //int fs = n / 0.02;              // Hz. Sampling frequency.
```

```
9     int i=0;                          // Cursor 1
10    int j= n/2. +1;                   // Cursor 2. The voltages must be 180 degrees shifted.
11    int dutycycle [510];              // Array that contains the values for the dutycycle.
12    // int dutycycle [n];             // It only works if the value is defined manually.
13
14    //.....SETUP.....
15    void setup () {
16
17    // PIN DEFINITION.
18    pinMode (9, OUTPUT);
19    pinMode (10, OUTPUT);
20
21    //.....DUTYCYCLE COMPUTATION.....
22    for (i=0; i<=n; i++)
23      {
24        dutycycle[i] = (int) ( (1 + sin (2.* PI *i/n ))* (TOP/2) + 0.5);
25      }
26
27    //..... TIMER 1: PWM.....
28    cli ( );                          // Deactivating global interruptions
29    TCNT1= 0;                         // Initialize counter value to zero.
30    TCCR1A = (1<<COM1A1) | (1<<COM1B1) | (1<<WGM11);
31     // Non-inverting, fast PWM MODE. Clear OC1A/OC1B on Compare match. Mode 14.
32     TCCR1B= (1<<WGM13) | (1<<WGM12) | (1<<CS10);
33     // Counts from the bottom to the top and then decreases. No prescaling.
34
35    // ICR1= (int) (16000000 / f_pwm)-1;
36    // Overflow value. It sets the resolution of the PWM.
37     ICR1=TOP;
38    // In case another PWM frequency is desired, this value must be changed.
39    OCR1A= dutycycle[i];               // Dutycycle for pin 9
40     OCR1B= dutycycle[j];              // Dutycycle for pin 10
```

```
41      //..................................... TIMER 2: INTERRUPTION.....................................................
42      TCCR2A=0;                          // Setting the entire registers to zero.
43      TCCR2B=0;
44      TCCR2A | = (1<<WGM21);         // Turn on Comparing Output Mode, non PWM.
45      TCCR2B | = (1<<CS21);           // Prescaler at 8.
46      // In case it is changed, the formula below must be changed to the new prescaler.
47      //OCR2A = (byte) ((16000000 / (fs * 8) ) - 1);
48      // Number of pulses it counts until the interruption is activated.
49      OCR2A= 78;
50      TIMSK2 |= (1 << OCIE2A); // Timer 2 Output Compare Match A Interrupt Enable
51      sei ( );                           // Global interruption activation
52      }  //setup
53
54      void loop ( )      {
55      }
56
57      //.....................TIMER 2 INTERRUPT SERVICE ROUTINE.....................
58      ISR (TIMER2_COMPA_vect) {
59      if (i<= n-1)
60      {
61      OCR1A= dutycycle [i];
62       i++;
63      }
64      else { i=0; } ;
65      if ( j<= n-1)
66      {
67       OCR1B= dutycycle[j];
68       j++;
69      }
70      else { j=0; }
 71     }  //ISR
```

Appendix C: Modes of Operation in Arduino.

This Appendix collects the Bit Description for the Waveform Generation in Arduino UNO. As explained before, there are 7 Modes of Operation available in Timer 0 and 2 and 15 on Timer 1. Reference: [21].

- **Timer 0:**

Table 15-8. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Notes: 1. MAX = 0xFF
2. BOTTOM = 0x00

Figure C.1: Waveform Generation Mode Bit Description for Timer 0 [21]

- **Timer 1:**

Table 16-4. Waveform Generation Mode Bit Description(1)

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1X at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX

Table 16-4. Waveform Generation Mode Bit Description(1) (Continued)

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1X at	TOV1 Flag Set on
13	1	1	0	1	(Reserved)	–	–	–
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Note: 1. The CTC1 and PWM11:0 bit definition names are obsolete. Use the WGM12:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

Figure C.2: Waveform Generation Mode Bit Description for Timer 1 [21]

- **Timer 2:**

Table 18-8. Waveform Generation Mode Bit Description

Mode	WGM22	WGM21	WGM20	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Notes: 1. MAX= 0xFF
2. BOTTOM= 0x00

Figure C.3: Waveform Generation Mode Bit Description for Timer 2 [21]

Appendix D: Design Documents

Element	Value	Units	Number of units	Reference Name
Capacitors	10	nF	5	
	100	nF	1	
	0.33	µF	1	
	0.47	µF	4	
	2.2	µF	2	
	22	µF	2	
	47	µF	1	
Inductances	470	µH	1	
Resistors	10	Ω	4	
	180	Ω	1	
	500	Ω	4	
	47	kΩ	4	
Diode			6	1N4004
Zener diode 15V			4	1N4728
MOSFET			4	IRF60B217
Driver			2	IR2110
Inverter gate			1	74LS04
Voltage regulator			1	LM7805C

Table D.1: Resume of the components of the circuit.

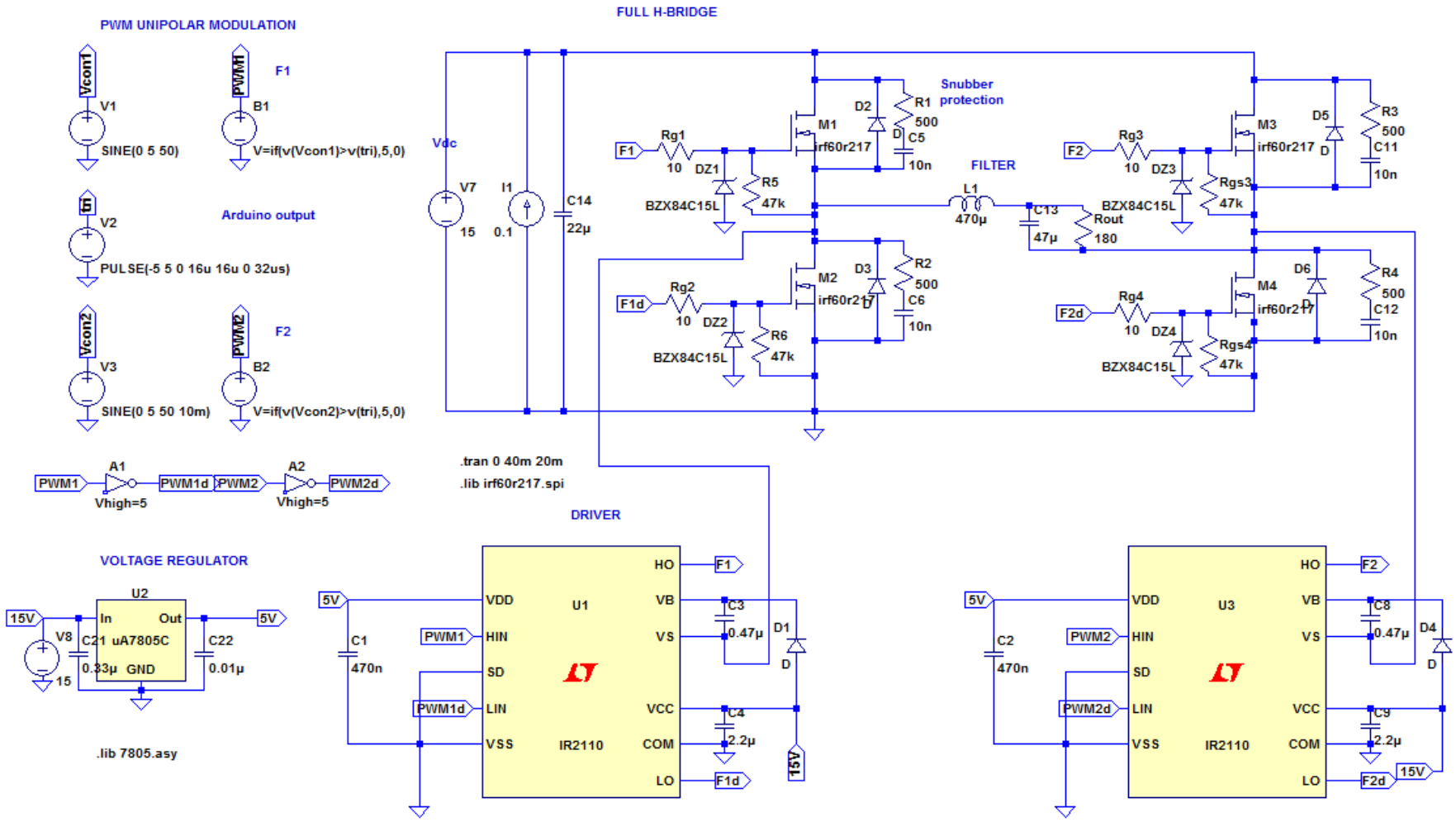


Figure D.1: Schematic with LTSpice software

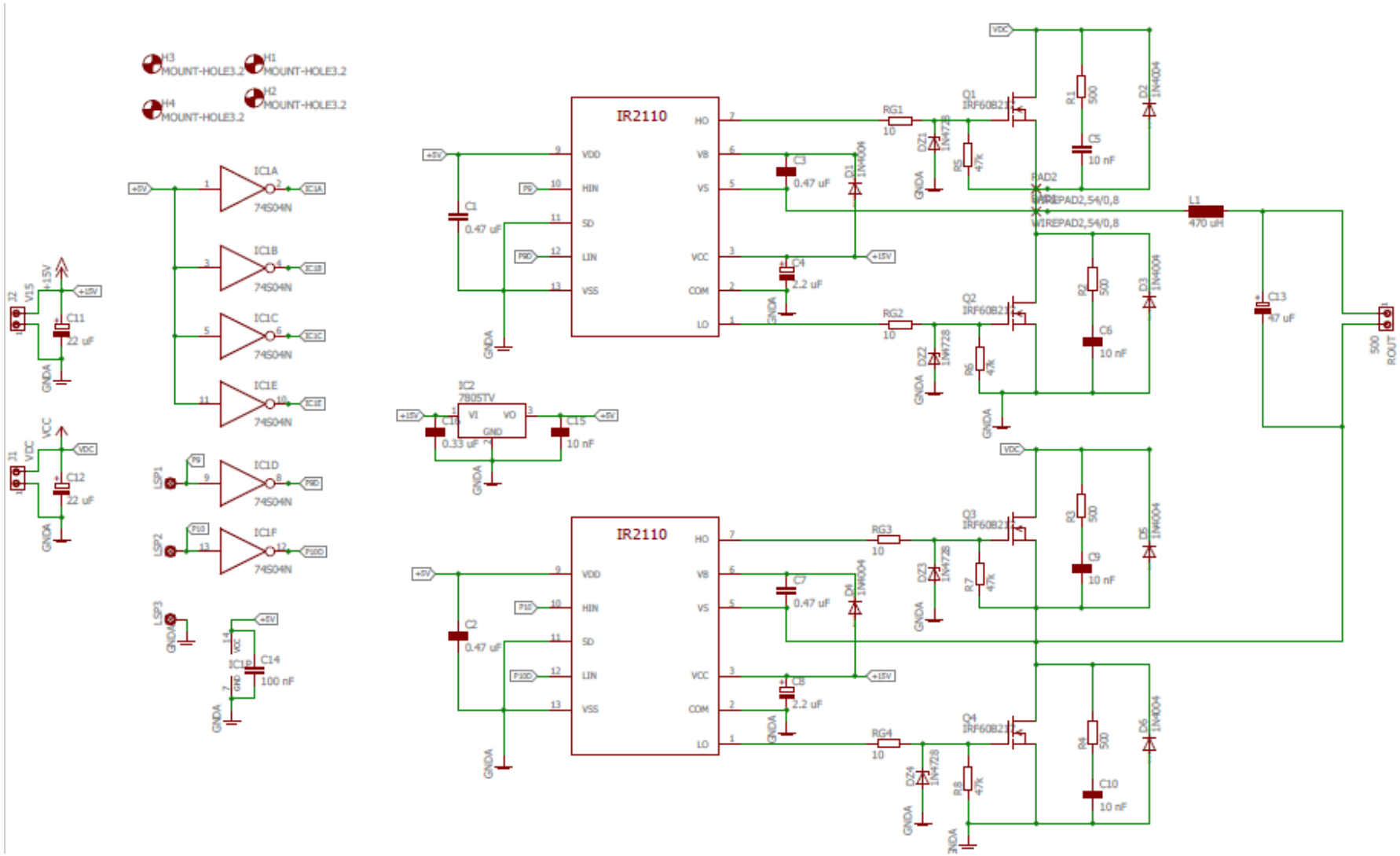


Figure D.2: Schematic with EAGLE software

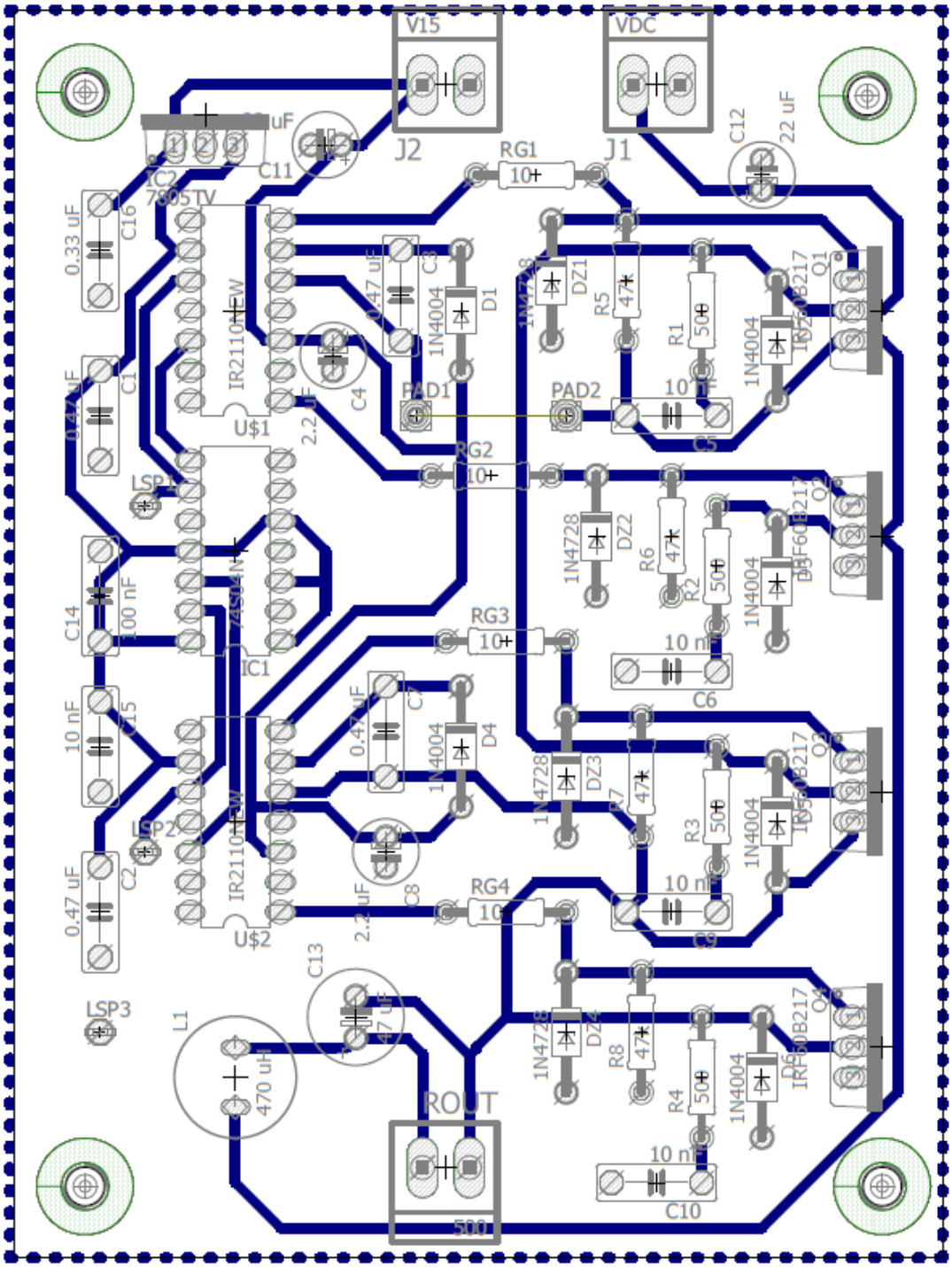


Figure D.3: Board design.

Size: 79.6 x 106.2 mm

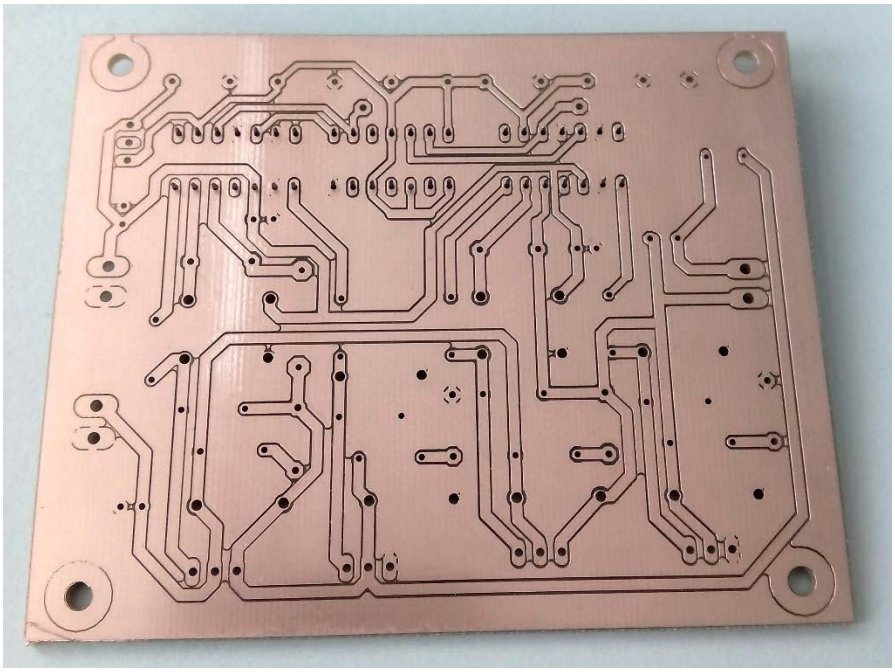


Figure D.4: Real PCB without elements. Bottom side

Figure D.5: Real PCB with elements