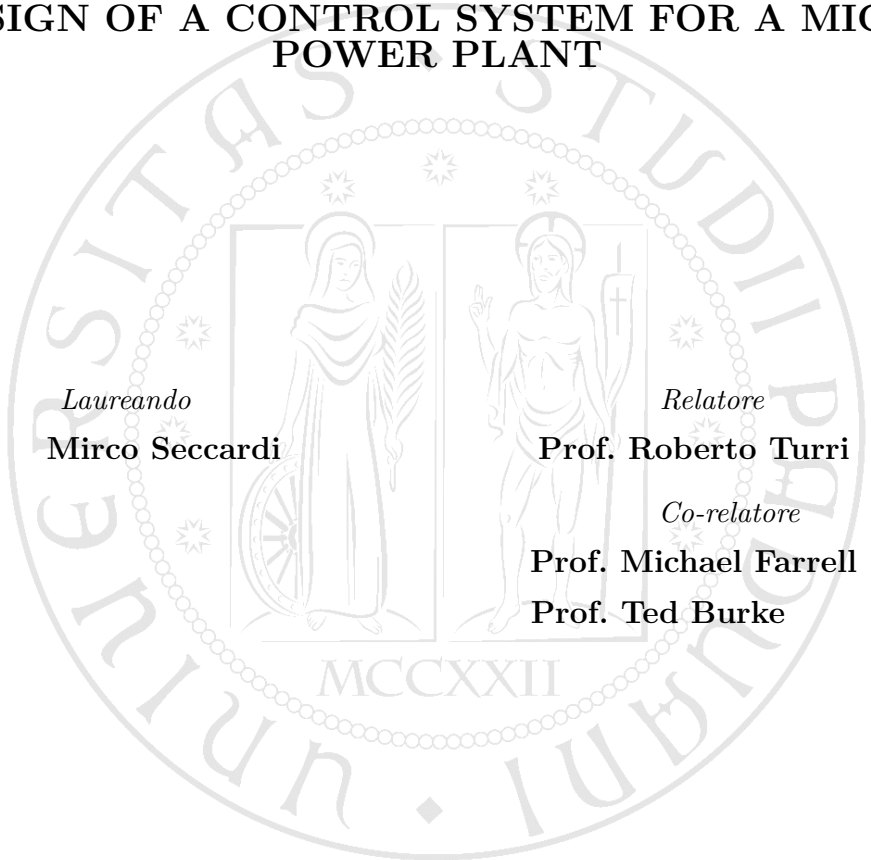
UNIVERSITÁ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA INDUSTRIALE

corso di laurea magistrale in ingegneria dell'energia elettrica

DESIGN OF A CONTROL SYSTEM FOR A MICRO POWER PLANT



Laureando

Mirco Seccardi

Relatore

Prof. Roberto Turri

Co-relatore

Prof. Michael Farrell

Prof. Ted Burke

in collaborazione con

Dublin Institute of Technology

ANNO ACCADEMICO 2016/2017

Table of Contents

List of Figures

List of Tables

Abstract

As energy is one of the prime factors for the economic growth, it is vital to sustain a modern economic and social development. Nowadays, the increase in energy demand and climate change problems are global issues. Renewable energy and energy efficiency offer solutions for reducing green house gas emissions and providing the requested energy to billions of consumers. The oil prices of traditional power plants are uncertain and can cause economic crisis that can last for decades in this industrialised world. The answer for this can be in part represented by a clean energy production through windmills, solar power panels and micro hydro power plants. Renewable and efficiency energy technologies allows providing electricity that is low-carbon, clean, safe, reliable and independent on fuel prices. In this paper a simulation of an hydro power plan is analysed showing how a production control could vary for example depending on factors that have zero impact in the fossil fuel demand and consumption.

Acknowledgements

I express my deepest sense of gratitude to Prof. Michael Farrell (head department of electrical engineering) for all the help he gave me (the assistance for the work in the laboratory, the burocracy I encountered during my erasmus period and much more). I am very thankful to Prof. Ted Burke, lecturer in robotics at the school of electrical engineering systems, for the keen interest and encouragement regarding the second part of my work. I acknowledge with thanks Prof. Roberto Turri, the university of Padua, the Dublin Institute of Technology (DIT) and the European Union which gave me the opportunity to study abroad. I also express gratitude to my family for all the support (especially to my grandmother who prepared for me the most succulent dishes), my Dublin friends (Azra, Angelo, David, Kevin, Giovanni, Giulia, Michela, Mino, Juan, Shane, Stephen) and not (Alberto, Billy, Bozza, Giulia, Kevin, Michael, Tosty and many others) and last but not least my lover Georgia who I shared all the pain and joy with and always will I.

1 Introduction

Nowadays there are many concerns about how the earth can continue to bear an increasing number of people leading the energy demand to higher levels and causing serious harm to the planet. In fact the energy that everyone needs can not only be taken from food and beverage since there are other daily necessities such as commuting by car or by public transport, charging smartphones and so on. Over these last 20 years, technology had done enormous steps both improving our lives and developing energy saving goods. However, all the amount of troubles that affect our ecosystem (sea level rising, ice caps melting, greenhouse effect and more) have to be dealt with. A first step could be to further enhance the efficiency and diffusion of cleaner electricity generation processes. In this report a small hydropower system will be examined showing how a small primary source can be employed in delivering power to the grid. Although this is a zero greenhouse gases emission technology, we need a supply of constant (or slightly variable) water for having the chance to implement this project. Where such type of primary source is not available, other types of energy are exploited such as solar radiation and wind force. In this project the small hydro turbine is simulated thanks to a DC motor directly coupled through common shaft, to the induction machine rotor. The asynchronous machine working thus as a generator, transform the mechanical input power into electrical power. Afterwards the electrical output is rectified and a DC/AC inverter is used for interfacing the right voltage and frequency between the generator and the grid. The project involve testing the asynchronous machine in order to gather necessary information about its components and characteristics. MATLAB/SIMULINK has been used for carry out simulations so as to acquire suitable and reliable control. Figure 1.1 offers a schematic overview about the project here studied.

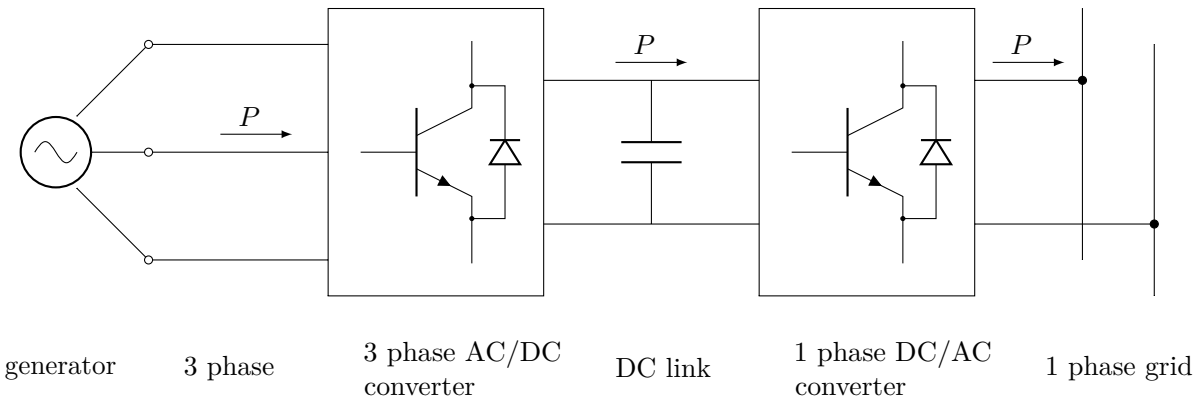


Figure 1.1: Electrical overview of the project

the generator feeds the three phase line with electrical power P (figure 1.1) passing through a three phase rectification where DC values are obtained. Afterwards, a DC/AC single phase inverter is implemented so the output can be injected to the single phase household power line. The reason of all these power conversions can be justified by the decoupling acquired between induction generator and grid which guarantees a constant RMS voltage V and frequency f operation. If the decoupling has not been done the situation would be as described in the subsection 2.2.1.

1.1 Project organization

To begin with, many reports, books and lecture notes has been consulted in order to explain in the best way possible the various characteristics of the induction machine and what processes we are involved in. The report continues with verifying induction machine theory thanks to MATLAB/SIMULINK models. The first schemes deal with motor parameters and the generator behaviour such as magnetizing curve, speed-torque graph and self excitation process which involves calculating the proper size of the capacitors in order to supply the required magnetizing current to the machine (section 2.3). Having carried out the digital trials, a real asynchronous machine is employed for comparing real trends with the simulated ones beforehand collecting data stored in chapter 4. All those data has been stored at the end of this report. Finally a control method of the system is designed focusing on delivering the same amount of power to the single phase grid in case of variable mechanical power input (chapter 6 and chapter 7). Such control is laid out because real variations of input power given by water usually happen and thus it is essential to deal with them and have a constant electric power output.

2 Induction machine

An *induction machine* (*asynchronous machine*) is an electric motor in which the electric current in the rotor needed to produce torque is obtained by electromagnetic induction from the magnetic field of the stator winding. An induction machine can therefore be made without electrical connections to the rotor (figure 2.1). The stator windings consist of insulated wire copper coils fixed into slots properly shaped onto the stator iron which is formed by thin sheets of steel in order to lessen eddy currents. There are many types of induction machines that differ to each other by rotor characteristic; the most commons are:

wound rotor It is the only induction machine which has windings both in the stator and in the rotor. Because of this and all the aftermaths that such system has (greater quantity of raw material and complex realization, wider and more reliable control level needed) it is the most expensive and the less employed in the micro hydro power systems. On the other hand, it plays an important role as for its use as DFIM (Doubly Fed Induction Machine) where the use of brushes ensure electrical power supply to the rotor windings.

squirrel cage It is the most common and has a simpler theory behind its employment. The rotor consists of a few alluminium bars that are short-circuited at each end by alluminium rings.

double squirrel cage It has two independent cages on the rotor. Bars of high resistance and low reactance are placed in the outer cage, and bars of low resistance and high reactance are placed in the inner cage. The outer cage has high $\frac{X}{R}$ ratio whereas the inner cage has low $\frac{X}{R}$. This affects the starting torque which will be larger in comparison with the other types of rotors.

The treatise in this chapter will consider the machine working as a motor unless otherwise specified.

2.1 Equivalent circuit

The equivalent single phase circuit of a three phase squirrel cage induction machine is depicted in figure 2.2 and in 2.3. It shows the various parameters of the machine such as its Ohmic losses and also other losses modeled by inductors and resistors. The copper losses are occurred in the windings so the winding resistance is taken into account. In addition, the windings has a dissipative inductance for which there is a voltage drop due to inductive reactance. Figure 2.2 shows the equivalent circuit considering stator and rotor as two physically different parties like in the reality. However, the equivalent single phase scheme of figure 2.3 is far more used,

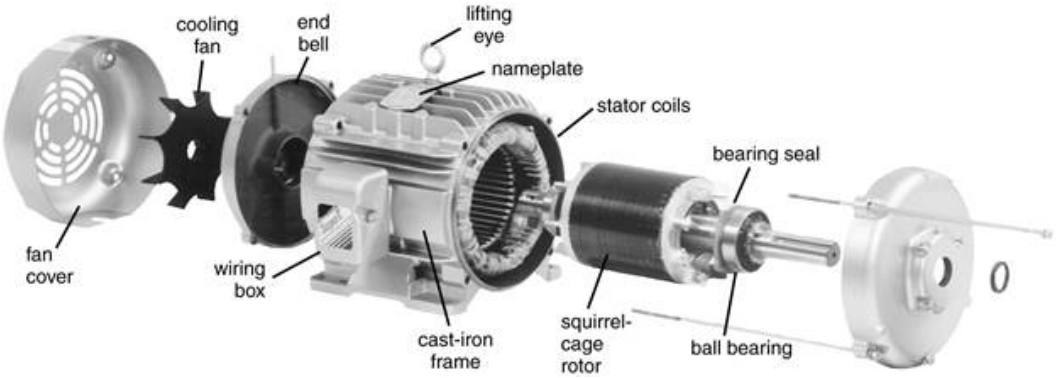


Figure 2.1: Exploded view of an asynchronous machine

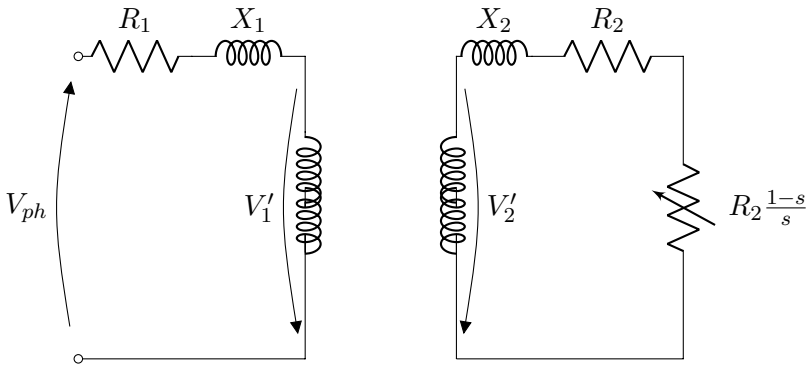


Figure 2.2: Mutual coupling beetwen stator and rotor

where there are rotor resistance R_{12}, rotor reactance X_{12} and $R_{12}\frac{1-s}{s}$ with referred to the stator windings follow the relation

$$R_{12} = R_2 \cdot t^2 \qquad X_{12} = X_2 \cdot t^2 \tag{2.1.1}$$

where

$$t = \frac{V_1'}{V_2'} \tag{2.1.2}$$

in figures 2.2 and 2.3 the labels stand for:

- V_{ph} phase voltage supplied to the terminals;
- R_1 stator winding resistance;

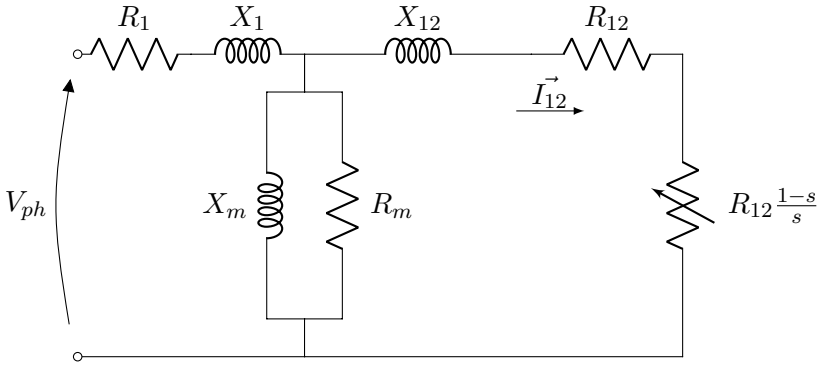


Figure 2.3: Equivalent single phase circuit

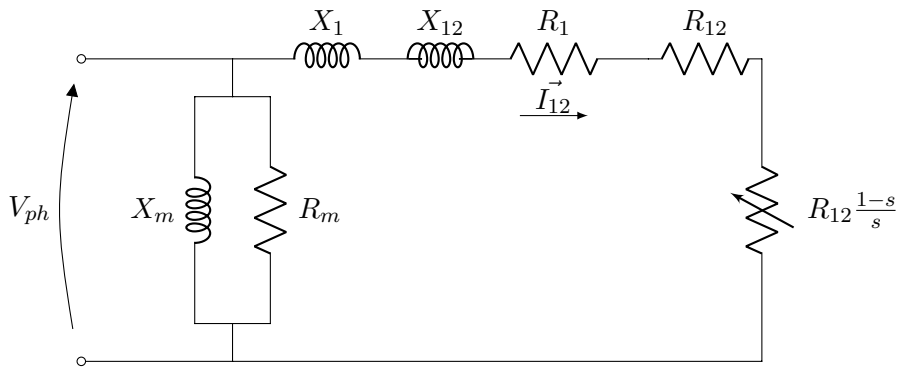


Figure 2.4: Semplified single phase equivalent scheme

- X_1 stator winding reactance;
- X_m magnetizing reactance which represents the flux linked beetwen stator and rotor;
- R_m core losses;
- V_1' back electromagnetic force at the stator;
- V_2' back electromagnetic force at the rotor;
- X_2 rotor winding inductance;
- R_2 rotor winding resistance;
- $R_2\frac{1-s}{s}$ resistance that represents the power required by the load;
- R_{12} rotor resistance reffered to stator windings;
- X_{12} rotor reactance reffered to stator windings;
- s slip factor well known thanks to the 2.3.4;
- $R = R_1 + R_{12}$ the equivalent resistance;
- $X = X_1 + X_{12}$ the equivalent reactance.

Since the stator losses have a negligible value (especially for higher rated power than this report face) the stator branch could be moved inside the shunt impedance in order to work with a simpler interpretation of the induction machine as 2.4 depicts.
In the circuit of figure 2.4 we can take into account these following considerations:

1. when the rotor speed n_r is equal to the synchronous speed n_0

$$s = 0 \quad \Longrightarrow \quad R_{12}\frac{1-s}{s} = \infty \tag{2.1.3}$$

this means that there is a small current value only in the shunt branch of figure 2.4 (I_m). This condition is only theoretically possible since there would always be mechanical losses which yield to have a rotor speed below n_0. However, this assumption is generally feasible during no load tests since air friction and bear ring would slightly affect n_r (figure 2.9);

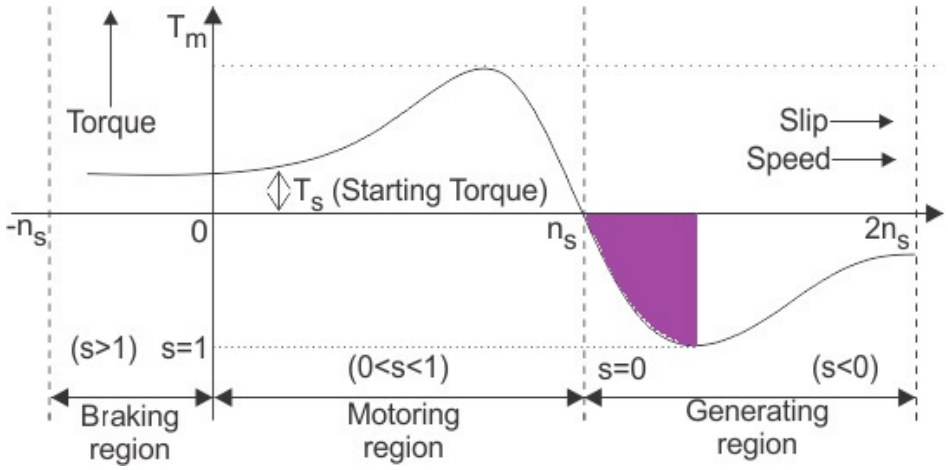


Figure 2.5: Generic speed-torque characteristic of the induction machine

2. in the case of $n_r = 0$

$$s = 1 \quad \Longrightarrow \quad R_{12}\frac{1-s}{s} = 0 \tag{2.1.4}$$

we obtain the evaluation of the shortcircuit losses since there is no load attached offering an electrical continuity in the circuit (figure 3.2);

3. Between these two brinks the slip factor varies moving the induction machine to different operation states

$$R_{12}\frac{1-s}{s} \neq 0 \quad \Longrightarrow \quad P_m = 3R_{12}\frac{1-s}{s}I_{12}^2 = T\omega_r = T\frac{2\pi n_r}{60} \tag{2.1.5}$$

where P_m is the mechanical power delivered by the shaft to the load having neglected mechanical power losses such as drag and friction. A generic characteristic can be seen in figure 2.5 where various regions are distinguished according to the s value.

2.2 Induction generator

When the rotor speed n_r is greater than the synchronous speed n_0 the machine works as a *generator* (figure 2.5). This can be obtained thanks to a prime mover which delivers mechanical power P_m to the generator shaft. After the electrical conversion, thanks to the magnetic coupling between rotor and stator, electrical power is delivered to the output terminals (figure 2.6). Consequently, we must have a mechanical input power P_m higher than P_{el}, oppositely at when we consider the induction motor operation. Stable regions are marked in purple in figure 2.5.

$$n_r > n_0 \quad \Longrightarrow \quad s = \frac{n_0 - n_r}{n_0} < 0 \tag{2.2.1}$$

$$P_m = T\omega_r = T\frac{2\pi n_r}{60} > P_{el} = 3V_{ph}I_1\cos\varphi \tag{2.2.2}$$

where φ is the phase between V_{ph} and I_1. This type of electric machine accepts variable and constant loads, starts either loaded or without load and is capable of continuos or intermittent operation. The squirrel cage rotor can be specifically designed to work with wind or hydro turbines, in other words with a larger slip factor, with more convenient deformation in the torque curve, winding sized to support higher saturation current, increased number of poles.

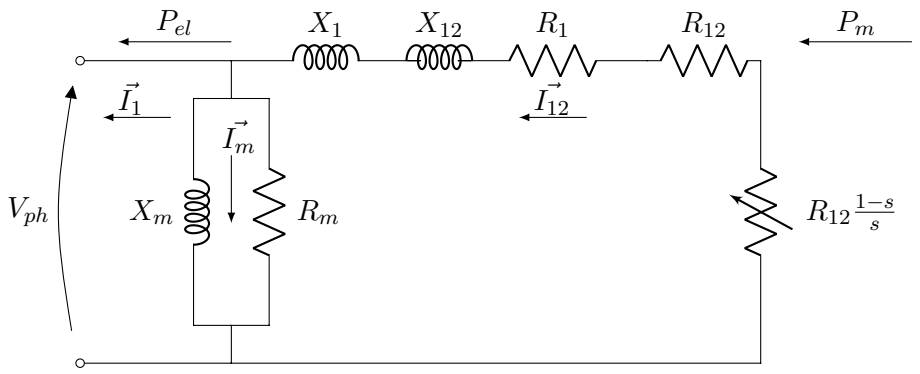


Figure 2.6: Induction generator equivalent scheme

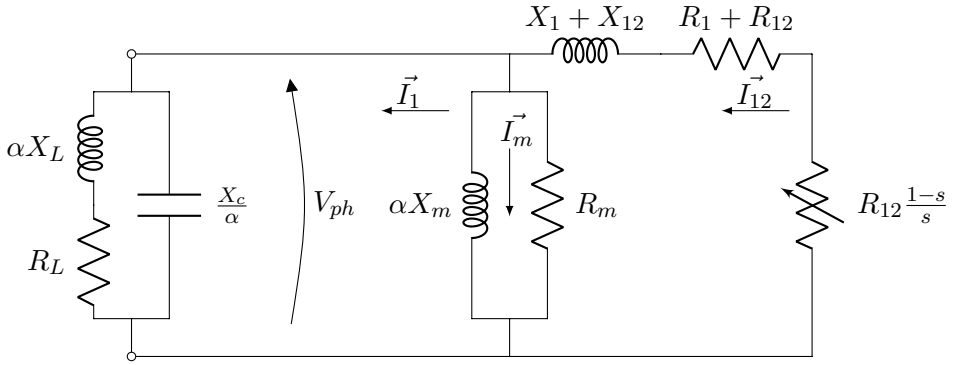


Figure 2.7: Equivalent semplified scheme of SEIG with bank of capacitor delivering power to the load

2.2.1 Standalone induction generator

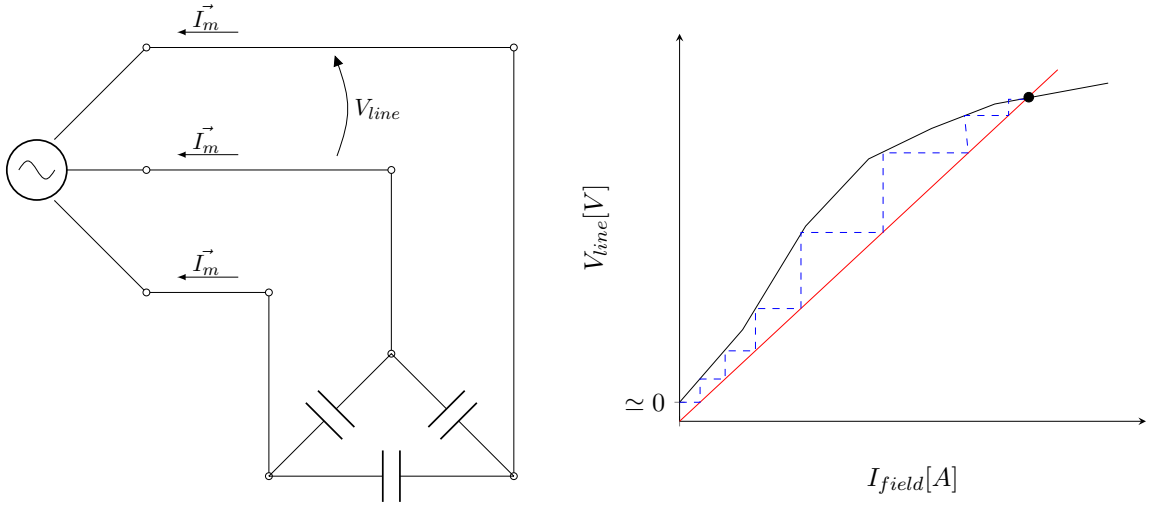
A bank of capacitor is connected for supplying the reactive power Q_{flux} required to create magnetic flux inside the generator (figure 2.8a). As a result, the system is named SEIG (Self Excited Induction Generator). With the electric load attached and thus with the standalone system on, the voltage V_{phase} and the frequency f depend entirely on the rotor speed n_r and on the circuital characteristics such as reactance X and resistance R. Assuming α as the ratio between the real frequency in steady state conditions and the desired frequency, the equivalent scheme is illustrated in figure 2.7. In order to know what the delivered power to the load is, we first must know the magnetizing curve (at no load) and by this, we can calculate the proper value of capacitance needed in the excitation process.

2.3 Self excitation process

This process is primarily used to build up a voltage at the generator terminals. All starts from the residual magnetism in the iron, that produces a small voltage. Consequently a capacitive current (delayed current) appears and produces an increased voltage that provokes a higher increase of the capacitive current, and so on, until the iron saturation of the magnetic field (figure 2.8b). The magnetizing curve is costumarily represented either by a polynomial or non linear expression as

$$L_m = a_0 + a_1 V_{ph} + a_2 V_{ph}^2 + a_3 V_{ph}^3 + a_4 V_{ph}^4 \tag{2.3.1}$$

$$X_m = \omega_{exc} L_m = \frac{V_{ph}}{I_m} = k_1 e^{k_2 I_m^2} + k_3 \tag{2.3.2}$$



(a) process at no load **(b)** graph building

Figure 2.8: Self excitation

without the residual magnetism the induction generator cannot produce self excitation. It is important to emphasize that is very difficult to lose residual magnetism completely; a minimum value always remains. In the case of complete loss, there are four commonly used techniques for recovering it:

- make the machine rotate at no load and a high speed until the residual magnetism is recomposed;
- use a battery to cause a current surge in one of the machine windings;
- maintain a charged high-capacity capacitor to cause a current surge as in the previous method;
- use a rectifier fed from the network to substitute for the battery in the second method.

A way for obtaining a more coherent magnetizing curve with the frequency-dependent nature of the induction motor parameters is to use a secondary machine coupled to its shaft. With this, a constant rotation is always guaranteed during the test. For operation as a standalone generator, it should be connected to a three-phase bank of capacitor in delta-connection so as to use the lowest capacitance for economical benefit. The capacitive reactance will be a straight line whose slope is

$$X_c = \frac{1}{\omega_{exc} C} = \frac{1}{2\pi f_{exc}} \tag{2.3.3}$$

where:

- ω_{exc} angular frequency in $\frac{rad}{s}$;
- f_{exc} frequency measured in $Hertz$.

2.3.1 Calculation of the capacitance level

First of all, it is essential to know the value of the magnetizing reactance of the induction machine. Thus a no load motor test is carried out. The equivalent single phase scheme (figure 2.3) can be approximate to (figure 2.9) since

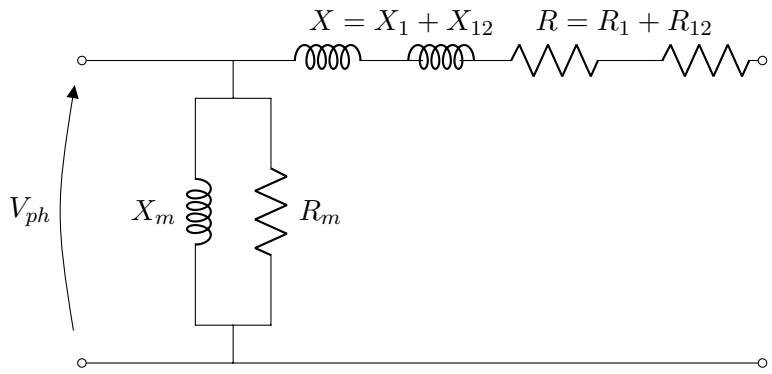
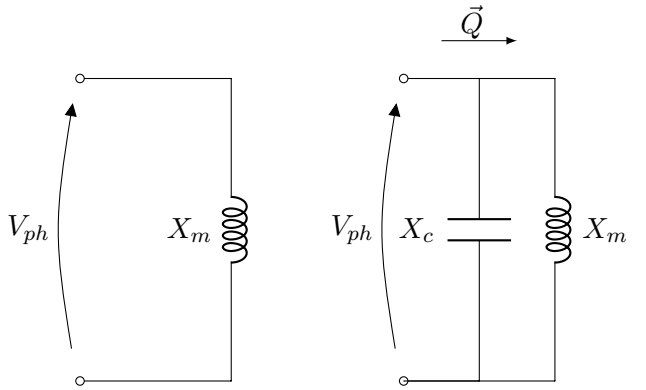


Figure 2.9: No load test equivalent scheme



(a) Final circuitry in no load test

(b) Self excited generator test

Figure 2.10: Semplified no load test equivalent schemes

$$n = 1500rpm \implies s = \frac{n_0 - n_r}{n_0} \simeq 0 \tag{2.3.4}$$

where:

- n_0 synchronous speed;
- n_r rotor speed.

A further step is to neglect resistance R_m since is always verified that $X_m << R_m$. Eventually the overall equivalent circuitry is reduced as depicted in figure 2.10a.
The task is to apply a capacitance which can guarantee the required inductive reactive power Q_{flux} for build up the desired voltage at the generator terminals (figure 2.10b).

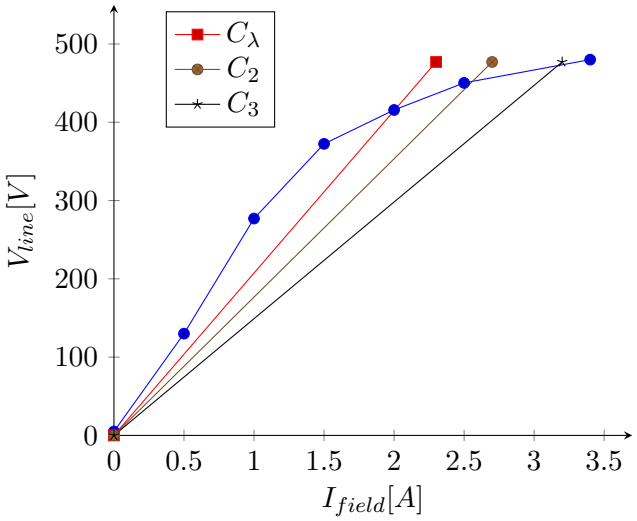
2.3.2 Trial generator model

Typing the saturation parameters in Simulink (figure 2.11b), for the preset model used in this project, onto the induction generator window and afterward doing no load test, we obtain

$$X_c = \frac{V_{ph}}{I_m} = \frac{V_l}{\sqrt{3}I_m} = \frac{415}{\sqrt{3} \cdot 2} = 120\Omega \tag{2.3.5}$$

$$C_\lambda = \frac{1}{\omega X_c} = \frac{1}{2\pi f X_c} = \frac{1}{2\pi 50 \cdot 120} = 26.5\mu F \tag{2.3.6}$$

(a) Characteristics intersection



(b) Saturation parameters

$I_{field}[A]$	$V_{line}[V]$
0	5
0.5	130
1.0	277
1.5	372.4
2.0	415.7
2.5	450.3
3.4	480

Figure 2.11: Magnetizing curve parameters and capacitance slope set in the trial model

knowing that $C_\lambda = 3C_\Delta$ the level of capacitance required in order to have $V_l = 415V$ is

$$C_\Delta = \frac{C_\lambda}{3} = \frac{26.5}{3} = 8.83\mu F \tag{2.3.7}$$

thanks to these computations the figure 2.11a is obtained. A delta connection for the capacitance is usually applied since there is a cost reduction due to lower level of capacitance needed (2.3.7) in order to reach the output voltage V_{line}. However, in this project we assume a wye connection between the C bank and the induction machine so as to have the conveniency to simplify the model to an equivalent single phase scheme for future calculations.
As we notice from figure 2.11a having $C_\lambda < C_2 < C_3$, the terminal voltage can be determined thanks to the size of the capacitors connected when the machine rotates at constant speed. We conclude that the greater is the capacitance value the higher is the voltage.

3 Induction machine laboratory tests

We now focus the attention to the different tests carried out in the laboratory machine in order to acquire a full knowledge of the theoretical approach explained in chapter 2. It has to be said that we have used *delta connection* for the machine windings and thus all the treatise is refered to *delta equivalent circuits*.

3.1 No load test

The circuit is illustrated in figure 3.1 where the movement of stator resistance R_1 and stator reactance X_1 behind the magnetised impedance $Z_m = X_m//R_m$ is allowed. This does slightly affect the values of the equivalent parameters obtained being narrowly away from what they are supposed to. Despite this, it is often considered a correct approach. Because of the proximity with the synchronous speed and since there is no load attached, the equation 2.3.4 can be taken as true leading to figure 2.9. During the test the measures in table 3.1 have been taken

$V_{line}[V]$	$I_{line}[A]$	$P[W]$	$Q[VAR]$	$S[VA]$	$\cos(\varphi)$	$n[RPM]$
220	2.85	178	1072	1088	0.16	1499

Table 3.1: Noted values during the no load test

the phase angle (angle between V_{ph} and $\vec{I}_{ph}$) is derived from the power factor $\cos(\varphi)$

$$\varphi = \arccos(\cos(\varphi)) = \arccos(0.16) = 1.41rad \quad \rightarrow \quad \varphi = 80.79^\circ \tag{3.1.1}$$

the impedance can be shown thanks to figure 3.1 where

$$\vec{z}_m = \frac{V_{ph}}{\vec{I}_{ph}} = \frac{V_{line}}{\frac{\vec{I}_{line}}{\sqrt{3}}} = \frac{220}{1.67\underline{/-80^\circ}} = 22.87 + \jmath 129.74\Omega \tag{3.1.2}$$

as a verification of R_m and X_m the powers P and Q are exploited. Indeed

$$R_m = 3\frac{V_{ph}^2}{P} = 3\frac{220^2}{178} = 815.73\Omega \tag{3.1.3}$$

$$X_m = 3\frac{V_{ph}^2}{Q} = 3\frac{220^2}{1072} = 135.45\Omega \tag{3.1.4}$$

having thus verified the 3.1.3 and 3.1.4, the impedance at no load becomes

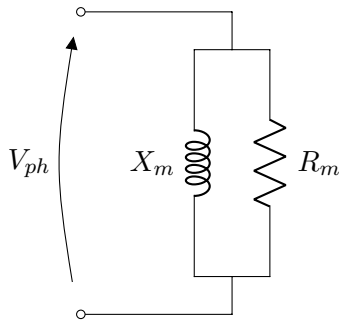


Figure 3.1: No load test equivalent circuit

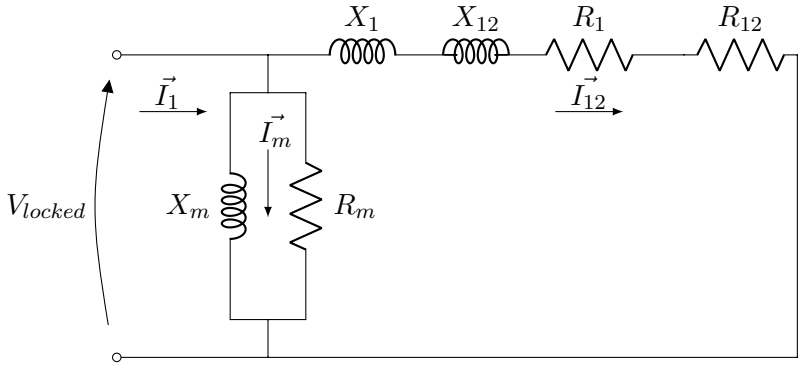


Figure 3.2: Equivalent scheme for locked rotor test

$$\vec{z}_{m_{theory}} = R_m//X_m = \frac{R_m \cdot \jmath X_m}{R_m + \jmath X_m} = \frac{815.73 \cdot \jmath 135.45}{815.73 + \jmath 135.45} = 21.89 + \jmath 131.82\Omega \simeq \vec{z}_m \quad (3.1.5)$$

3.2 Locked rotor test

This evaluates the remaining impedance of the equivalent scheme depicted in figure 2.4 where, thanks to 3.1.3 and 3.1.4, we can calculate the current $\vec{I}_{12}$ and the short circuit impedance $\vec{z}_{ss}$. It has to be said that the rated current is reached with a small voltage (V_{locked}) because of the fact that the rotor is not able to rotate. As a consequence

$$s = 1 \rightarrow R_{12}\frac{1-s}{s} = 0 \quad (3.2.1)$$

and the equivalent circuit becomes as shown in figure 3.2. The values captured from the power analyser, voltmeter, ammeter are gathered in table 3.2. Like the previous no load treatise, the phase angle is now equal to

$$\varphi = \arccos(\cos(\varphi)) = \arccos(0.67) = 0.83rad \quad \rightarrow \quad \varphi = 47.93^\circ \quad (3.2.2)$$

the all impedance of the circuit in figure 3.2 can be calculated

$$\vec{z} = \frac{V_{locked}}{\frac{\vec{I}_{line}}{\sqrt{3}}} = \frac{58.9}{2.86\underline{/-47.93^\circ}} = 13.8 + \jmath 15.28\Omega \quad (3.2.3)$$

we can remove, thanks to the 3.1.5, the magnetised impedance in order to obtain a close-to-reality level about the short circuit impedance $\vec{z}_{ss}$. Proceeding forward

$V_{locked}[V]$	$I_{line}[A]$	$P[W]$	$Q[VAR]$	$S[VA]$	$\cos(\varphi)$	$n[RPM]$
58.9	4.96	341.8	373.5	506.3	0.67	1499

Table 3.2: Noted values during the locked rotor test

Figure 3.3: DC machine present in the lab and employed in the project as prime mover

$$\vec{z} = \frac{\vec{z_m} \cdot \vec{z_{ss}}}{\vec{z_m} + \vec{z_{ss}}} \quad \rightarrow \quad \vec{z_{ss}} = -\frac{\vec{z} \cdot \vec{z_m}}{\vec{z} - \vec{z_m}} \tag{3.2.4}$$

which gives

$$\vec{z_{ss}} = -\frac{(13.8 + \jmath 15.28)(22.87 + \jmath 129.74)}{(13.8 + \jmath 15.28) - (22.87 + \jmath 129.74)} = 17.43 + \jmath 15.94\Omega \tag{3.2.5}$$

as a verification of the values R and X shown altogether in 3.2.5, thanks to the power read on the various measurement instruments over the test

$$P = I_{line}^2 R = I_{line}^2 (R_1 + R_{12}) \quad \rightarrow \quad R = \frac{P}{I_{line}^2} = \frac{341.8}{4.96^2} = 13.89\Omega \tag{3.2.6}$$

$$Q = I_{line}^2 X = I_{line}^2 = (X_1 + X_{12}) \quad \rightarrow \quad X = \frac{Q}{I_{line}^2} = \frac{373.5}{4.96^2} = 15.18\Omega \tag{3.2.7}$$

that are more similar to the components calculated in 3.2.3. This incompatibility with $\vec{z_{ss}}$ in 3.2.5 can be caused by the level of voltage reached in the locked rotor test that varies the level of the magnetizing inductive reactance as we can see in the saturation curve built in figure 2.11a. As a result the values in 3.1.3 and 3.1.4 cannot be considered valid in the locked rotor test here undertook. Therefore it is more reliable to work with 3.2.3 instead of carry out all the treatise with the aid of 3.2.5.

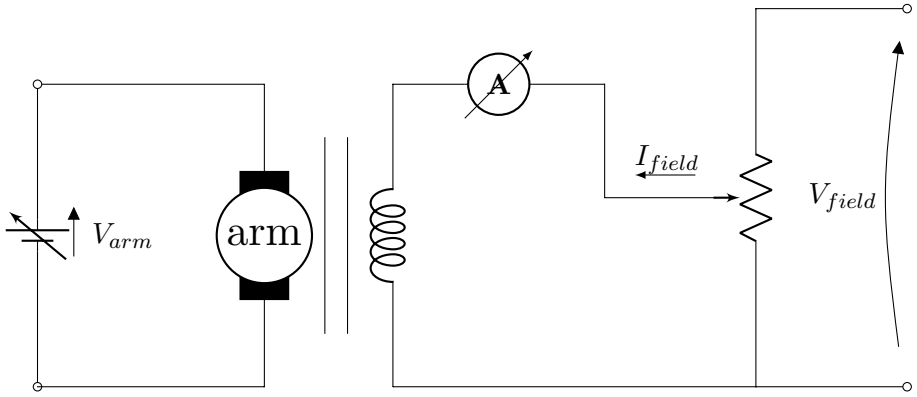


Figure 3.4: Equivalent scheme for a generic DC machine

3.3 Magnetising curve

Dragging the induction machine at the synchronous speed of $1500RPM$ with the aid of a DC machine (whose plaque is depicted in figure 3.3) along with varying the terminal voltage of the IM, yields to have a dependence between V_{line} and I_{line} that does not comply the linear behaviour as someone would think. This is due to the *saturation curve* intrinsically present in the machinery. First of all, as a refresh, a DC machine works with 2 DC windings respectively laid onto the stator (*armature winding*) and the other onto the rotor (*field winding*). We can have three different methods in regards to the field current supply:

- *parallel excitation* where the field and armature windings are under the same voltage leading in $V_{arm} = V_{field} = V_{dc}$;
- *series excitation* in which the same current pours into the field winding as well as in the armature winding obtaining $I_{field} = I_{arm} = I_{dc}$;
- *independent excitation* all of the 2 systems has its own supply (figure 3.4).

Using the armature variable DC voltage source (set at $189V$ in figure 3.4) and injecting the right amount of field current (0.7A), the synchronous speed is quickly obtained. As a consequence, we can vary the voltage means of the induction machine. All the measurements of V_{line} and I_{line} are mingled in table 3.5a. The resulting graph is built in figure 3.5b. We can see the typical pace of a $V(I)$ characteristic where the kneel starts to appear around $V_{line} \simeq 150V$ detouring the trend from the straight line present until such value. During the test, also measures above the rated voltage of $220V$ has been taken and the final current has a value that is over the half rated current (equal to $8A$). This has been done in order to have a complete description for this specific machine.

3.4 Speed-Torque characteristic

Many values has been logged over the speed-torque characteristic test because they give an idea of what the asynchronous machine requirements are in terms of power and current while it works both as a motor and as a generator. The test is carried out maintaining a constant value of $V_{line} = 220V$ and in the meantime setting V_{arm} and I_{field} such that the synchronous speed of $1500RPM$ is achieved along with a zero torque production. Afterwards, the real test can start varying the level of load applied to the IM thanks to the common drive shaft between it and the DC machinery. A change in I_{arm} (obtained varying V_{arm}) leads into a load affecting the rotor speed and consequently the induction machine mechanical power output.

(a) Excitation values logged during the test

$V_{line}[V]$	$I_{line}[A]$
25.4	0.26
49.5	0.46
75	0.71
99.7	0.94
125.3	1.24
149.7	1.53
175.5	1.89
199.8	2.31
225.2	3.02
240.6	3.63
251	4.21

(b) Scope of the saturation curve

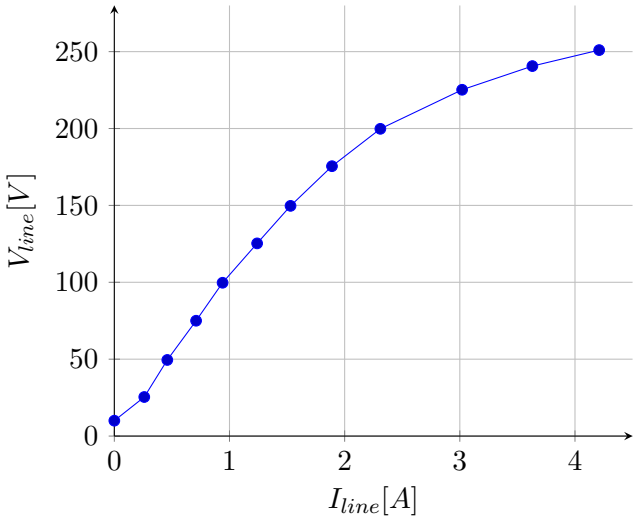


Figure 3.5: Magnetising curve construction

T $[Nm]$	n $[RPM]$	V_{line} $[V]$	I_{line} $[A]$	P $[W]$	Q $[VAR]$	S $[VA]$	$\cos(\varphi)$	I_{arm} $[A]$
0	1500	220	2.82	221	1046	1068	0.28	0.5
0.35	1492	220	2.83	212	1059	1076	0.2	0.1
2.5	1470	220	3.03	562.6	1010	1157	0.48	1.7
3.35	1462	220	3.2	700	997	1223	0.57	2.2
4.15	1453	221	3.45	832.6	1024	1319	0.63	3
4.65	1446	220.6	3.6	923	1021	1375	0.67	3.15
5	1442	220.3	3.7	985.2	1017	1415	0.7	3.45
5.59	1435	220.2	3.91	1090	1021	1490	0.73	3.8
6.09	1428	220	4.1	1183	1021	1563	0.76	4.1
6.60	1420	220.5	4.3	1285	1034	1649	0.78	4.6
7.07	1413	220.9	4.5	1363	1048	1715	0.79	4.9

Table 3.3: Values of motor speed-torque characteristic

All the measurements are reported in table 3.3 and in 3.4, whereas the complete speed-torque characteristic is shown in figure 3.6. The RMS values cannot be negative but here we change its sign automatically when we talk about I_{line} in the generator operation. It is also verified a higher peak for the IM torque when it runs as a generator; the explanation is given by the equation 4.3.5. Commenting the figure 3.6, we do not have a proper shape of the characteristic near the synchronous speed. This is down to the losses always present in the machine that are not negligible while the machine runs at $n \simeq n_0$ consuming a remarkable part of the active power flow. Indeed, examining what happens in the first dotted point in the generator region we have according to table 3.4, a positive demand of power since the generated power at this conditions is not enough even for covering the Joule losses. The same thing happens in the motor characteristic. We can not see the complete characteristic as in figure 2.5 because we end up in the unstable region.

T $[Nm]$	n $[RPM]$	V_{line} $[V]$	I_{line} $[A]$	P $[W]$	Q $[VAR]$	S $[VA]$	$\cos(\varphi)$
-0.47	1500	220.7	2.85	100	1088	1093	0.09
-2.20	1515	221.5	-3	-168	1135	1152	0.14
-3.37	1525	221.4	-3.2	-319	1205	1244	0.26
-4.34	1533	220.3	-3.5	-476	1251	1337	0.35
-5.5	1547	220.9	-3.83	-621	1324	1459	0.42
-6.4	1549	220.8	-4.12	-735.1	1399	1580	0.47
-7.15	1555	221	-4.36	-823	1443	1676	0.5
-7.9	1563	220.4	-4.6	-920.6	1483	1745	0.52
-8.52	1568	220.7	-4.8	-996.1	1547	1829	0.54
-9.16	1572	220.8	-5.1	-1082	1616	1940	0.55

Table 3.4: Values of generator speed-torque characteristic

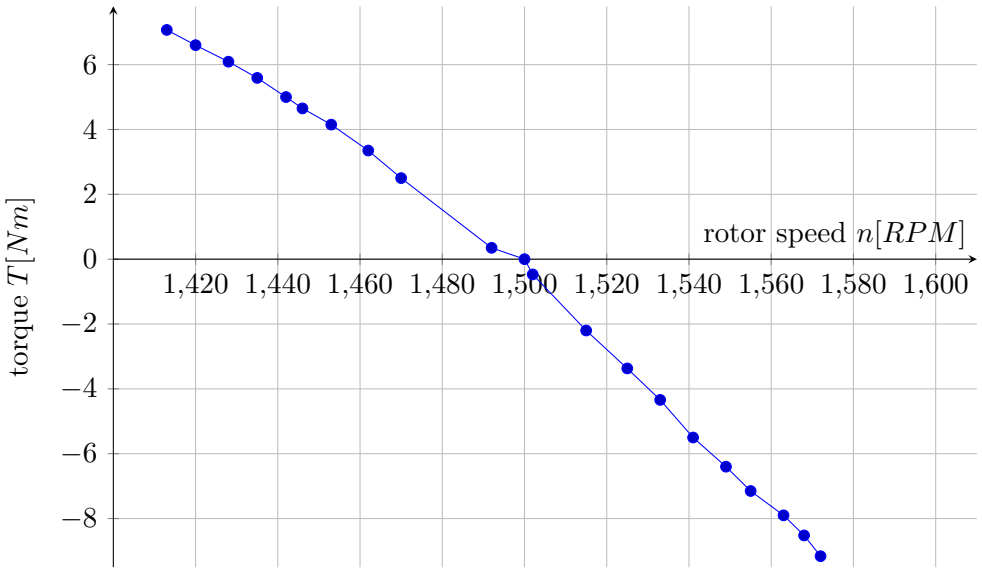


Figure 3.6: Complete speed-torque characteristic

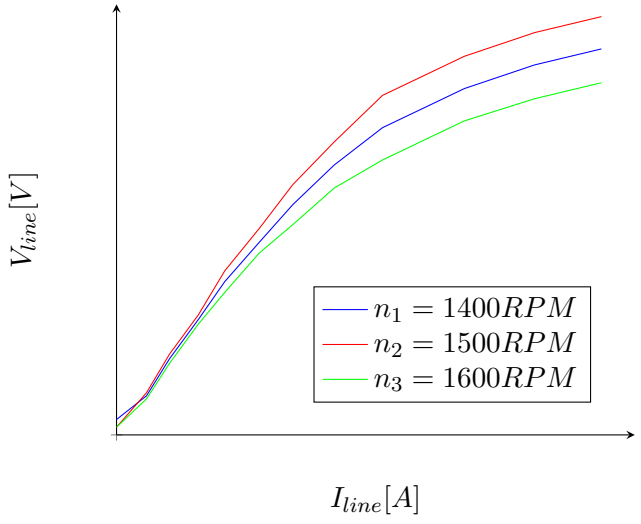


Figure 3.7: Generic saturation dependence on rotor speed

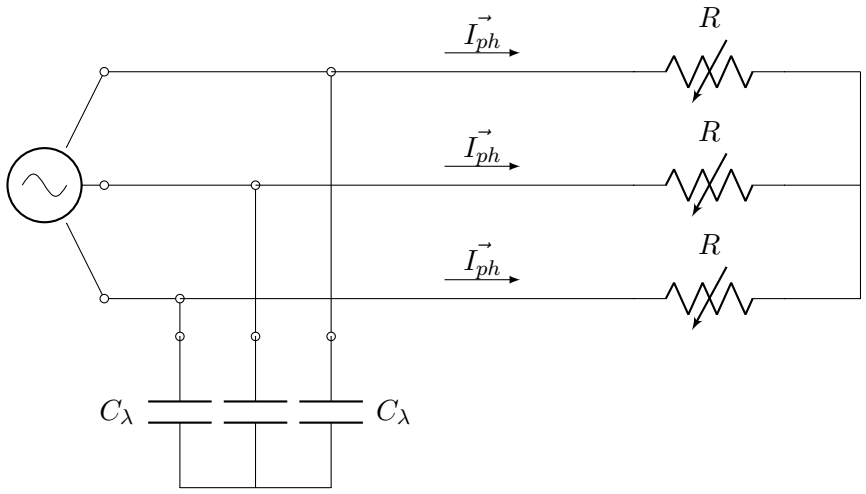
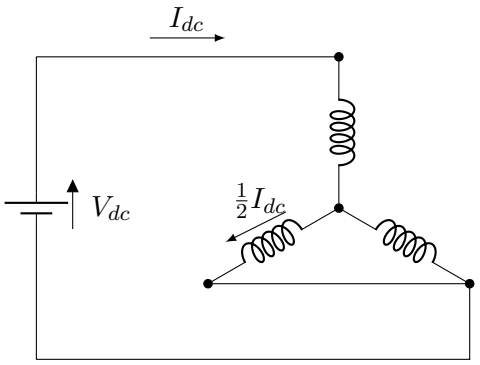


Figure 3.8: Circuit in the SEIG. both the C_λ and load are star connected

(a) With star connection windings



(b) With delta connection windings

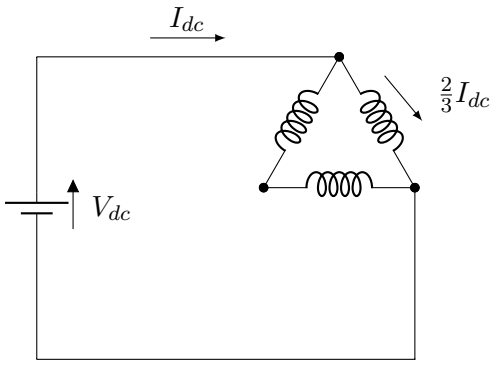


Figure 3.9: DC Flash poles connections available

3.5 Self Excited Induction Generator

The test is carried out dragging the IM over the synchronous speed as well as connecting a bank of capacitors in parallel to the circuit. We keep to use the DC machine for giving to the Self Excited Generator the required speed and we waive the grid means for reaching the desired voltage thanks to the reactive power coming from the capacitance. It has to be said that all the test is carried out with a star capacitor connection. Consequently we have C_λ as a capacitance value and the circuitry adopted is shown in figure 3.8. In this part, the level of capacitance is equal to $30\mu F$ and the aim is to see how the power drawn into the load changes maintaining the n constant and varying P_{load} thanks to the variable resistor.

3.5.1 Residual magnetism and flash poles

Before the test starts, we have to assure that there is residual magnetism in the induction generator windings. If it is not so (simply verified by running the self excited process without having an increase in voltage means), *flash poles* has to be realised. One of the methods for obtaining these flash poles is represented by feeding the asynchronous machine with a DC current in all the three phases regardless whether there is the same amount of current in each phase. Possible circuits (depending on the windings connection) are illustrated in figure 3.9.

T $[Nm]$	P_{shaft} $[W]$	V_{line} $[V]$	I_{line} $[A]$	P_{el} $[W]$	Q_{el} $[VAR]$	S $[VA]$	$\cos(\varphi)$
1.94	315	264	0	0.7	/	/	/
3.34	548	254.8	0.536	-235.7	-5.5	236	1
4.6	750	245.1	1.045	-442.7	-13	442.1	1
5.63	920	234	1.523	-615.6	-19.2	617.5	1
6.6	1080	220	2.04	-778	-23	776	1
7.2	1174	200	2.5	-876.8	-24	874.9	1

(a) Values for $n_1 = 1560RPM$

T $[Nm]$	P_{shaft} $[W]$	V_{line} $[V]$	I_{line} $[A]$	P_{el} $[W]$	Q_{el} $[VAR]$	S $[VA]$	$\cos(\varphi)$
1.82	296	257	0	0.7	/	/	/
3.18	512	249	0.52	-224	-5.2	224	1
4.36	700	239.6	1.007	-417.3	-21	418.3	1
5.44	872	228.8	1.483	-586	-18	586.5	1
6.38	1023	214.9	1.983	-737	-23	737	1
6.66	1073.8	184.9	2.5	-798	-21.8	800	1

(b) Values for $n_2 = 1540RPM$

T $[Nm]$	P_{shaft} $[W]$	V_{line} $[V]$	I_{line} $[A]$	P_{el} $[W]$	Q_{el} $[VAR]$	S $[VA]$	$\cos(\varphi)$
1.7	266.7	248	0	0.7	/	/	/
2.98	470	240	0.5	-209.8	-4.7	209.8	1
4.13	654	230.5	1.01	-393.5	-11	394.4	1
5.3	841	217.7	1.542	-583	-17.8	581.8	1
6.08	960	200.5	2.03	-705	-20.6	704.8	1
6.1	961	178.3	2.3	-714	-19.4	715.1	1

(c) Values for $n_3 = 1510RPM$

Table 3.5: Values collected during the test carried out at three different speeds

3.5.2 VI generator characteristic

We run the generator with three different rotor speeds above the synchronous one of $1500RPM$ just for simulating the behaviour in the grid even though it is a micro island operation, delivering the produced electric power P_{el} straightaway to the variable resistor load. Respectively the speed adopted are $n_1 = 1560RPM$, $n_2 = 1540RPM$, $n_3 = 1510RPM$. All the values are collected in table 3.5a, 3.5b, 3.5c and the RMS current value is now left positive even though the direction of the current is from the generator to the load. It is seen that the more P_{el} is asked by the load, the more voltage drop we will have. The speed of the generator affects the maximum output power achieved and even the overall performance: indeed, if the maximum demanded power conditions are examined for the all three operations

$$\eta_a = \frac{|P_{el_{max_a}}|}{P_{shaft_{max_a}}} = \frac{876.8}{1174} = 74.68\% \tag{3.5.1}$$

$$\eta_b = \frac{|P_{el_{max_b}}|}{P_{shaft_{max_b}}} = \frac{798}{1073.8} = 74.3\% \tag{3.5.2}$$

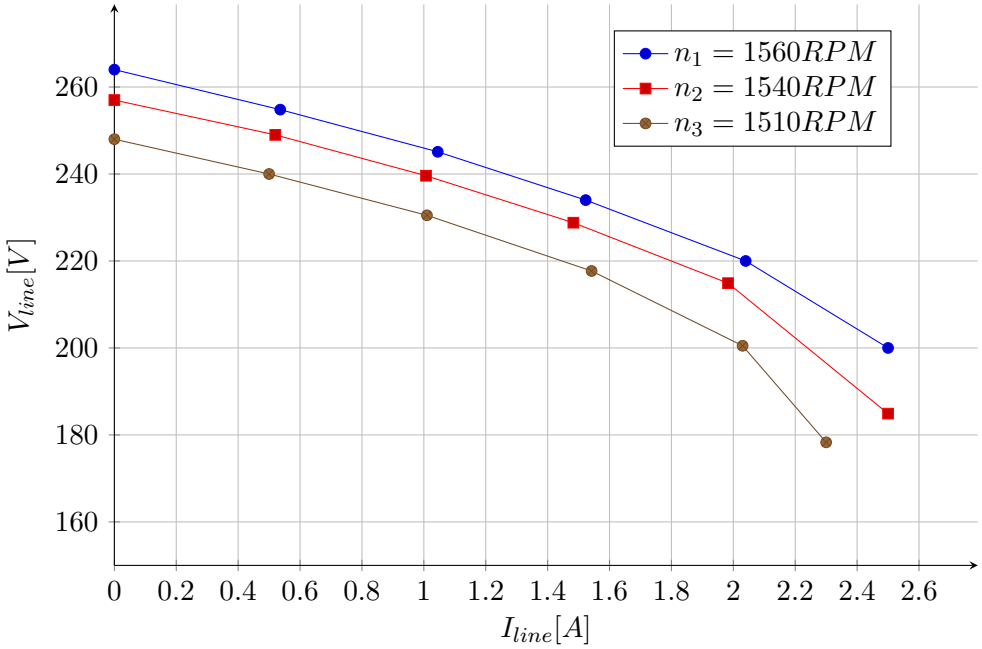


Figure 3.10: Table 3.5 graphically

$$\eta_c = \frac{|P_{el_{max_c}}|}{P_{shaft_{max_c}}} = \frac{714}{961} = 74.2\% \tag{3.5.3}$$

this is down to the stator and rotor resistance of the generator that, since R_1 and R_{12} are subject to higher currents (moving from test a to test c) their resistance value increase due to the temperature growth that naturally appears as a consequence of the current flow. Another value that depends on the speed with which the test is done, is represented by the line voltage V_{line}: in fact, the test at $1560RPM$ (table 3.5a) starts with the highest voltage rate for then decrease by greater steps during the following 6 measurements. In test b (table 3.5b) and c (table 3.5c) the initial voltages are far less than what are in the first speed value attempt. This is due to the different magnetising curve encountered for every speed taken into account (figure 3.7). The higher is the speed, the greater the values of voltages reached by the machine are. The graph in figure 3.10 shows the different paces of $V_{line} - I_{line}$ characteristics that are obtain in regards to the different speeds adopted. As depicted in figure 3.10 the trends follow a linear relationship up to values of current about $2A$. Afterwards the linearity is lost since the unstable generator operation (figure 2.5) begins to appear.

4

Real and linear machine behaviour

This chapter will treat what the main differences between a linear behaviour and a machine with saturation can be. A comparison then between the simulate linear behaviour (thanks to the MATLAB/SIMULINK environment) and the real one taken in the laboratory regarding the real machine described in chapter 3 will be shown. We consider in this chapter the machine working as a motor unless otherwise specified.

4.1 Magnetizing curve

For the knowledge of the induction machine values the most common means can be summarised in:

1. *no load characteristic/magnetizing curve*;
2. *self excitation process*;
3. *speed-torque characteristic.*

The model adopted is illustrated in figure 4.1 and it is formed by 2 induction motor blocks that represent a linear behaviour (upper model) and a real one with the presence of saturation (lower model). After having collected the several data needed for the graphic representation in a script file, different paces can be built as shown in figure 4.2. As it was explained in section 2.3, the magnetizing curve or saturation curve determines the terminal voltage for a given magnetizing current through the windings. It is necessary to obtain accurate knowledge about the magnetizing curve of an induction machine so that the magnetizing inductance can be updated according to different flux references for either efficiency optimization or field weakening control (not treated in this report). As previously cited, there are two trends to be studied and collected namely induction machine linear behaviour and in saturation operation. The detailed values of I_{phase} and the V_{line} steps are collected in table 4.1a. The figure 4.2 shows how the values collected create a different slope after a specific voltage $V_{line} \simeq 220V$. It is worth citing that in the entire report the preset model $n^{\circ} = 15$ has been used composed by preset values equal to:

- nominal power $5.4HP = 4kW$;
- nominal voltage $400V$;
- nominal frequence $50Hz$;
- nominal speed $1430RPM$.

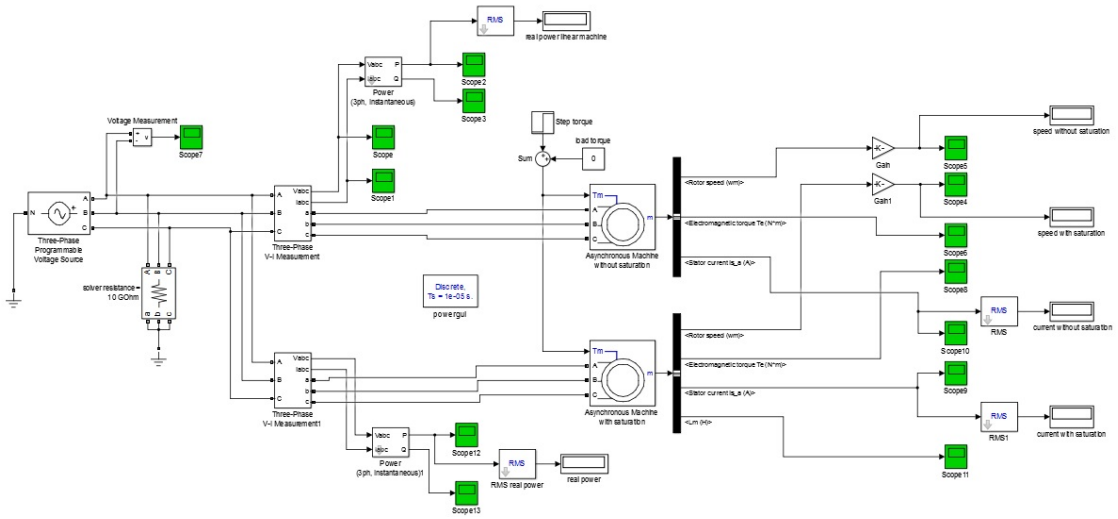


Figure 4.1: SIMULINK Model exploited for simulations

V_{line} $[V]$	**linear** I_{phase} $[A]$	**saturated** I_{phase} $[A]$
8.66	0	0
86.6	1.1	1.1
138.5	1.45	1.61
190.525	1.97	2.13
220	2.27	2.5
294.4	3.04	4.21
346.4	3.57	5.28
398.4	4.11	6.36
450.3	4.65	7.5

(a) Magnetising curve

speed $[RPM]$	**torque** $[Nm]$
1318	59
1336	55
1360	50
1375	45
1391	40
1431	26.7
1450	19
1474	10
1498	0

(b) Speed-torque characteristic

Table 4.1: Values detected for the knowledge of the induction machine

In addition the box "*Simulate saturation*" has to be ticked and the proper saturation parameters to be typed which correspond to the values of the red line in figure 4.2. The values gathered in table 4.1 and figure 4.2 have been reported in proportion to what found in the saturation curve in section 3.3. The real data could not be added to the preset model due to the different nominal rates.

4.2 Self excitation process

The rate of the excitation capacitance needed in order to work at $V_{line} = 220V$ with no load is

$$X_{exc} = \frac{V_{line}}{I_{phase}} = \frac{220}{2.5} = 88\Omega \tag{4.2.1}$$

$$C_{\Delta} = \frac{1}{2\pi 50 \cdot X_{exc}} = \frac{1}{2\pi 50 \cdot 88} = 3.61 \cdot 10^{-5} F = 36.1\mu F \tag{4.2.2}$$

$$C_{\lambda} = 3C_{\Delta} = 3 \cdot 3.61 \cdot 10^{-5} = 10.83 \cdot 10^{-5} F = 108.3\mu F \tag{4.2.3}$$

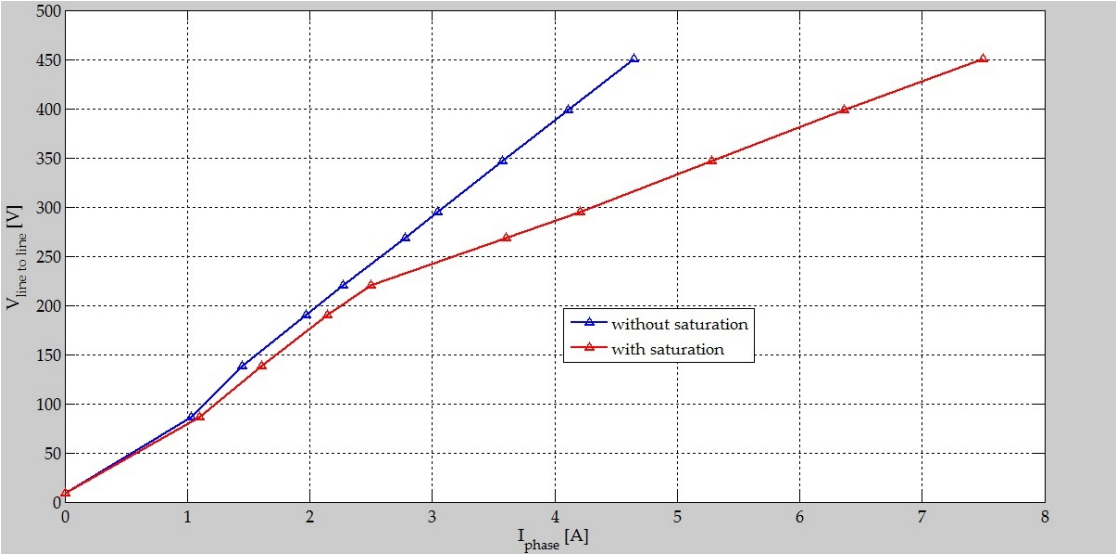


Figure 4.2: Different magnetizing curves

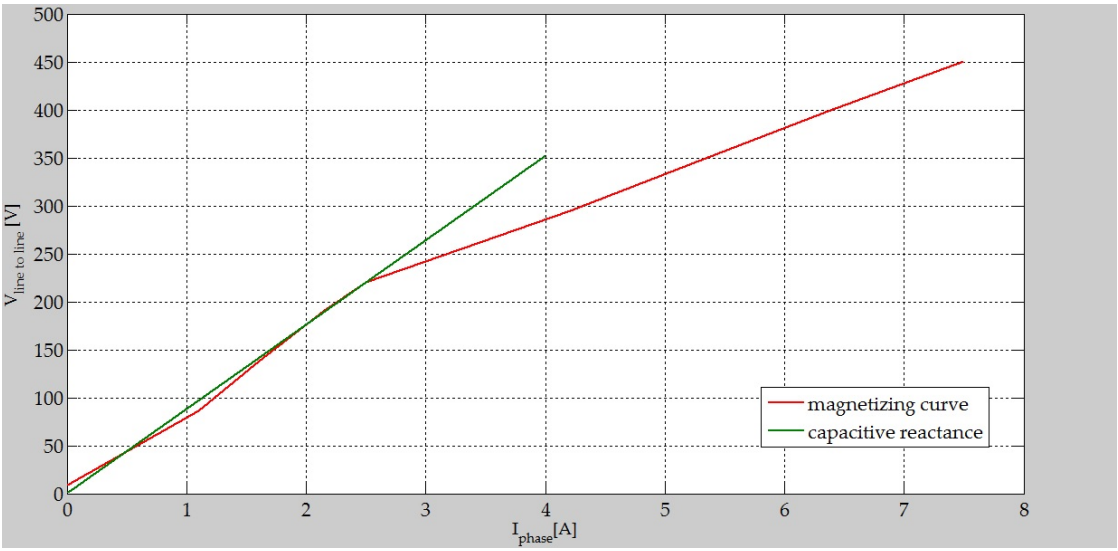


Figure 4.3: Magnetising and capacitive characteristics crossing

Although in figure 4.3 there are multiple crossing points before when it is due because of a none perfect saturation curve, it is demonstrated the effectiveness of 4.2.2. In fact

$$V_{line} = X_{exc} I_{phase} = \frac{1}{2\pi f C_{\Delta}} I_{phase} = \frac{1}{2\pi 50 \cdot 3.61 \cdot 10^{-5}} 2.5 \simeq 220V \tag{4.2.4}$$

a zoom is given by figure 4.4.

4.3 Speed-Torque characteristic

A torque-speed curve shows how a motor torque varies along with different conditions of load. All the values are collected in detail in table 4.1b. Only the part of the motor region has been here studied, but the generator characteristic is almost symmetrical containing a higher peak negative torque magnitude due to the fact that it is a function of the stator resistance. Indeed, if we examine the equivalent scheme in figure 2.4, an equivalent expression of the torque can

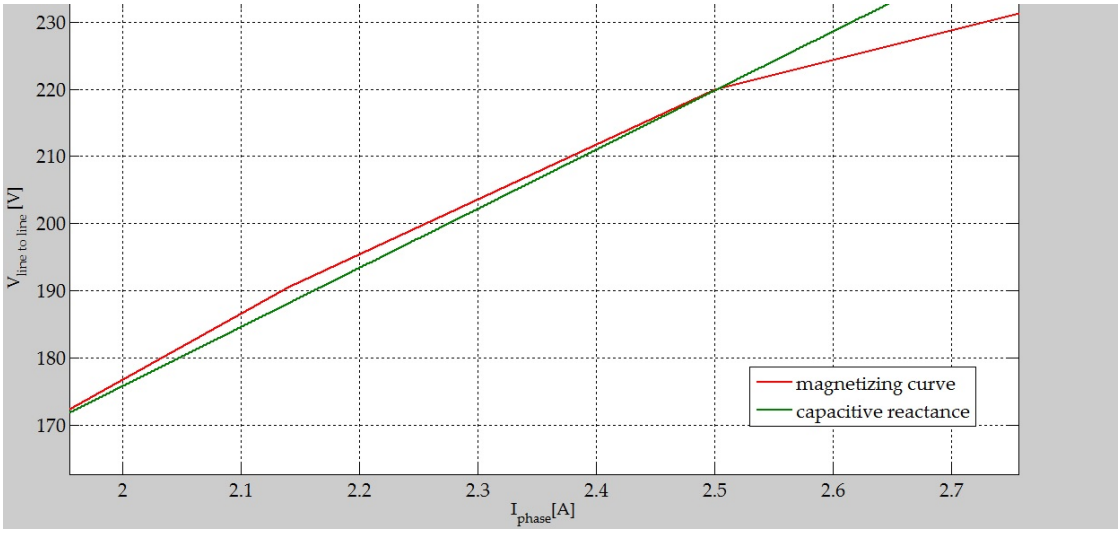


Figure 4.4: Magnetizing and capacitive characteristics crossing in detail

be

$$T = 3\frac{R_{12}}{s \cdot \omega_0} I_{12}^2 = \frac{R_{12}}{s \cdot \omega_0} \frac{{V_{ph}}^2}{(R_1 + \frac{R_{12}}{s})^2 + X^2} = \frac{{V_{ph}}^2}{\omega_0 \cdot Z(\frac{Z \cdot s}{R_{12}} + \frac{R_{12}}{s \cdot Z} + 2\frac{R_1}{Z})} \tag{4.3.1}$$

where

$$Z^2 = R_1^2 + X^2 \tag{4.3.2}$$

we know that the maximum torque is reached when $\frac{dT}{ds} = 0$ that brings to

$$s = \frac{R_{12}}{Z} = s_M \tag{4.3.3}$$

we can then write for the motor

$$T = \frac{{V_{ph}}^2}{2 \cdot \omega_0 \cdot Z(1 + \frac{R_1}{Z})} = T_M \tag{4.3.4}$$

and for the generator

$$T' = -\frac{{V_{ph}}^2}{2 \cdot \omega_0 \cdot Z(1 - \frac{R_1}{Z})} = T_m \tag{4.3.5}$$

the motor characteristic is shown in figure 4.5 in which we can assume that

$$T_m \simeq 18Nm \tag{4.3.6}$$

4.3.1 Performance

In the condition of maximum torque required, the power drawn to the motor by the grid is

$$P_{mech} = T_M \cdot \omega_M = T_M \frac{2\pi n_{T_M}}{60} = 18\frac{2\pi \cdot 1320}{60} = 2488.1W \tag{4.3.7}$$

as for the current coming out from the 3 phase electrical source we have

$$\hat{I}_{ph} = 13.5A \rightarrow I_{ph} = \frac{\hat{I}_{ph}}{\sqrt{2}} = \frac{13.5}{\sqrt{2}} = 9.54A \tag{4.3.8}$$

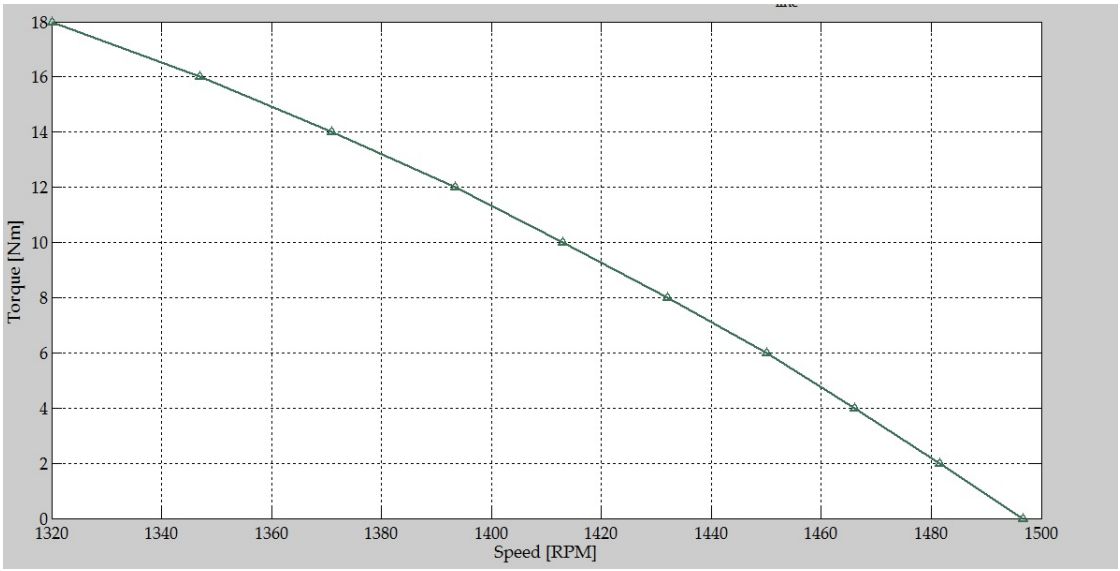


Figure 4.5: Speed-Torque motor characteristic

meanwhile the power factor is given by

$$\tan(\varphi) = \frac{Q_{3ph}}{P_{3ph}} = \frac{1600}{3278} = 0.488 \rightarrow \varphi = \arctan(0.488) = 26.01^\circ \tag{4.3.9}$$

having a $V_{ph} = 127V$ we can verify the P_{3ph} and Q_{3ph}

$$P_{3ph} = 3V_{ph}I_{ph}\cos(\varphi) = 3 \cdot 127 \cdot 9.54\cos(26.01^\circ) = 3266.6W \tag{4.3.10}$$

$$Q_{3ph} = 3V_{ph}I_{ph}\sin(\varphi) = 3 \cdot 127 \cdot 9.54\sin(26.01^\circ) = 1593.9VAR \tag{4.3.11}$$

consequently the performance of the induction machine is

$$\eta = \frac{P_{mech}}{P_{3ph}} = \frac{2488.14}{3266.6} = 76.18\% \tag{4.3.12}$$

which means that

$$P_{loss} = P_{3ph} - P_{mech} = 3266.6 - 2488.14 = 778.5W \tag{4.3.13}$$

is dissipated into the stator resistance R_{stat} and in the rotor resistance R_{rot}: indeed if we examine the stator losses we have

$$P_{R_{stat}} = 3R_{stat} \cdot I_{ph}^2 = 3 \cdot 1.405 \cdot 9.54^2 = 383.6W \tag{4.3.14}$$

taking into account the absence of a 3 phase system in the rotor squirrel cage

$$P_{R_{rot}} = 3R_{rot} \cdot (I_{ph_{rot}})^2 = 3 \cdot 1.395 \cdot \left(\frac{12.9}{\sqrt{2}}\right)^2 = 348.2W \tag{4.3.15}$$

that gives

$$P_R = P_{R_{stat}} + P_{R_{rot}} = 383.6 + 348.2 = 731.8W \tag{4.3.16}$$

and an error equivalent to

$$\varepsilon = \frac{|\Delta P|}{P_{loss}} = \frac{|P_{loss} - P_R|}{P_{loss}} = \frac{778.5 - 731.8}{778.5} = 6\% \tag{4.3.17}$$

s	θ_e	I_{ph_a}	I_{ph_b}	I_{ph_c}	φ_a	φ_b	φ_c
1	0	0	0	0	0	0	0

Table 4.2: Initial condition quantities

s	θ_e	I_{ph_a}	I_{ph_b}	I_{ph_c}	φ_a	φ_b	φ_c
0	0	1	1	1	0	120	240

Table 4.3: Initial conditions for the generator

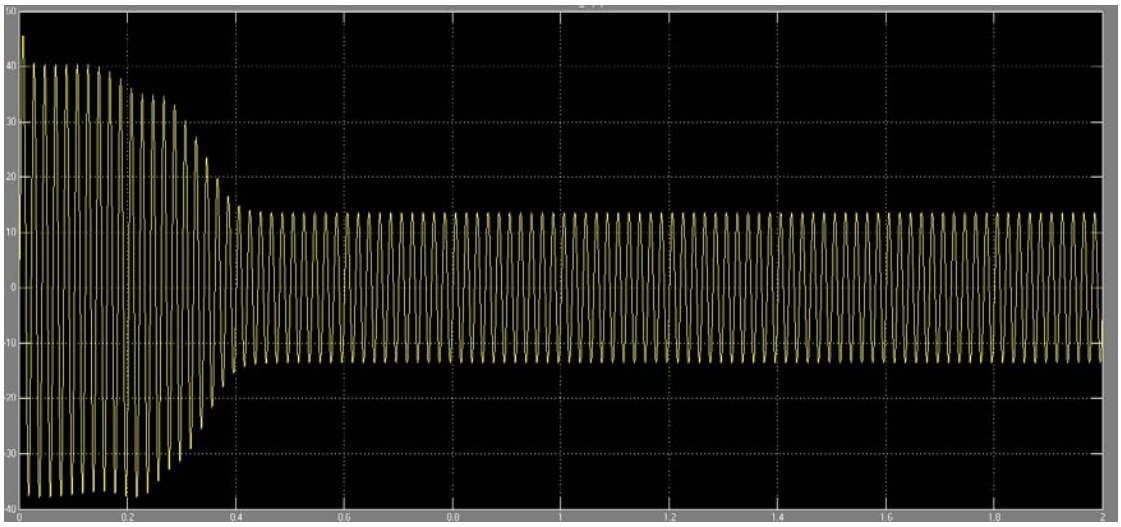
4.3.2 Waveforms

Figure 4.6a shows that the pace of phase a of the stator current in the saturated machine is, from the beginning to $0.4s$, affected by a transient behaviour moving from the initial conditions to steady state values. This initial transient affects all the other quantities such as rotor speed, active and reactive power, magnetising inductance and so on as depicted in figure 4.6. All this is down to the initial conditions of the "*induction machine*" block made by 8 values and collected in table 4.2 which respectively mean:

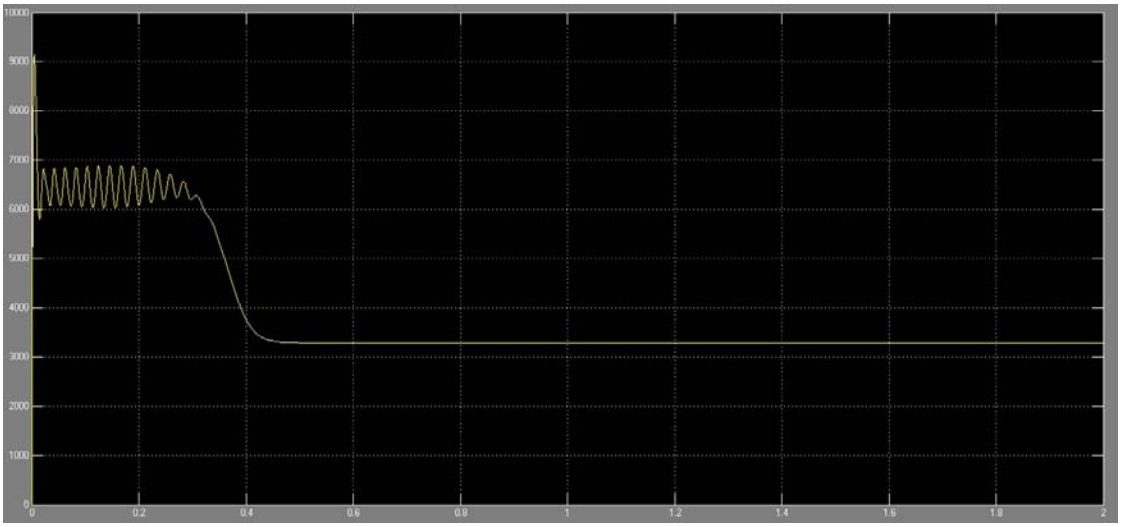
1. ***slip*** s that corresponds to 1 and consequently leads to a starting rotor speed of $0RPM$;
2. ***bias*** θ_e is the electrical angle equal to $\theta_e = p\theta_m$ where p is equal to pole pairs (must be inserted in *deg*);
3. I_{ph_a}, I_{ph_b}, I_{ph_c} magnitudes of equivalent 3-phase rotor currents;
4. φ_a, φ_b, φ_c phase angles of equivalent 3-phase rotor currents (must be inserted in *deg*).

as a result, if we modify these initial conditions (table 4.2) shrinking the gap between steady state and starting quantities, a smaller transient time will be achieved. The 3-phase programmable voltage source is not affected by any transient since it imposes the voltage the model works with. Finally different initial conditions for the generator have to be set, in fact we need the IM running since the model, working as a generator has to have a prime mover. Furthermore this is an ideal machine which does not model remnant flux in the iron core. This is required to develop a back electro magnetic force. We replicate this by establishing a d/q axis flux via initial conditions. The initial conditions must then include an initial stator current, for example $1A$. So table 4.2 is modified reaching table 4.3.

(a) Stator current I_{ph_a}



(b) Active power delivered by the grid



(c) Rotor speed

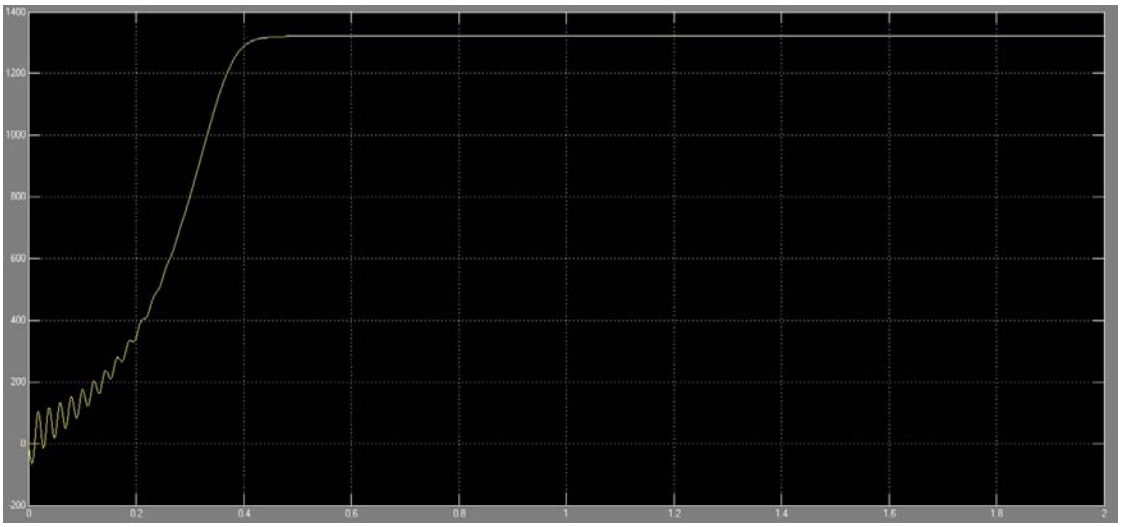


Figure 4.6: Scopes regarding the induction motor with saturation

5 Induction machine of the AC/DC/AC control

We will introduce the induction machine block used for the close loop schemes illustrated in the chapters ahead. This will play an important role since we take this machine as the model for setting and creating the future control. First of all, as described in section 4.1, the major characteristics of the IM has to be found.

5.1 Magnetizing curve

The model is equal with what depicted in figure 4.1 except for the fact that saturation parameters as well as V_{ph} has been changed once and for all. Having carried out numerous trials and collected all the data with different V_{ph} as gathered in table 5.1, the figure 5.1 shows what the magnetising curves are for both linear and real behaviour. At the first glimpse, we notice something theoretically impossible which is the fact that the alleged real behaviour reaches higher voltages than the linear machine as for constant current values. This can be considered acceptable in our case since we do not have any real correspondence with a physical induction generator and we want only to have a system in which our future control can work properly. If a real situation is faced, the saturation values will be widely changed.

5.2 Self excitation process

The right level of capacitance for the self excitation process has to be found: we know that for

$$V_{linetoline} = 400V \rightarrow I_{phase} = 1.8A \tag{5.2.1}$$

and the excitation reactance is equal to

$$X_{exc} = \frac{V_{linetoline}}{I_{phase}} = \frac{400}{1.8} = 222\Omega \tag{5.2.2}$$

finding an excitation capacitance that corresponds to

$$C_{\Delta} = \frac{1}{2\pi 50 X_{exc}} = \frac{1}{2\pi 50 \cdot 222} = 1.434 \cdot 10^{-5} F = 14.34\mu F \tag{5.2.3}$$

$$C_{\lambda} = 3C_{\Delta} = 3 \cdot 1.434 \cdot 10^{-5} = 4.315 \cdot 10^{-5} F = 43.15\mu F \tag{5.2.4}$$

lower than what we found in 4.2.2 and 4.2.3 even though a higher voltage is reached. This is due to the saturation pace this machine has which surges up to $\simeq 300V$ and slowly rise

V_{line} $[V]$	**linear** I_{phase} $[A]$	**saturated** I_{phase} $[A]$
8.66	0	0
86.6	1.01	0.32
138.5	1.46	0.61
190.525	1.97	0.74
220	2.27	0.81
294.4	3.04	1.09
346.4	3.57	1.37
398.4	4.11	1.8
450.3	4.65	2.54

(a) Magnetising curve

speed $[RPM]$	**torque** $[Nm]$
1330	60
1349	55
1368	50
1380	42
1399	40
1436	26.7
1450	19
1477	10
1490	0

(b) Speed-torque

Table 5.1: Values detected of the IM for the complete control architecture

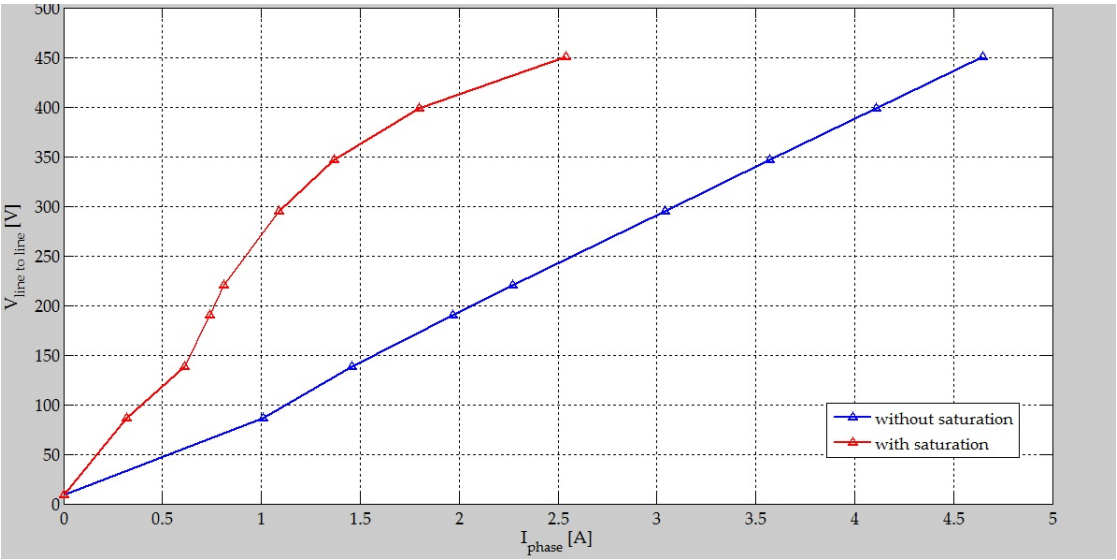


Figure 5.1: Comparison of the magnetising curves of the new "*induction machine*" block

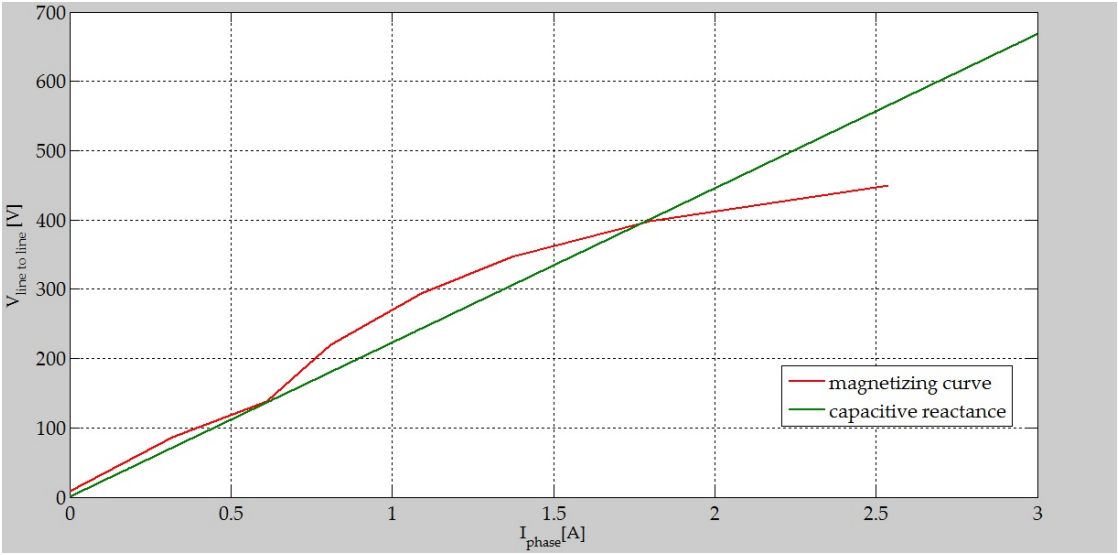


Figure 5.2: Clear intersection between capacitive reactance and saturation curve

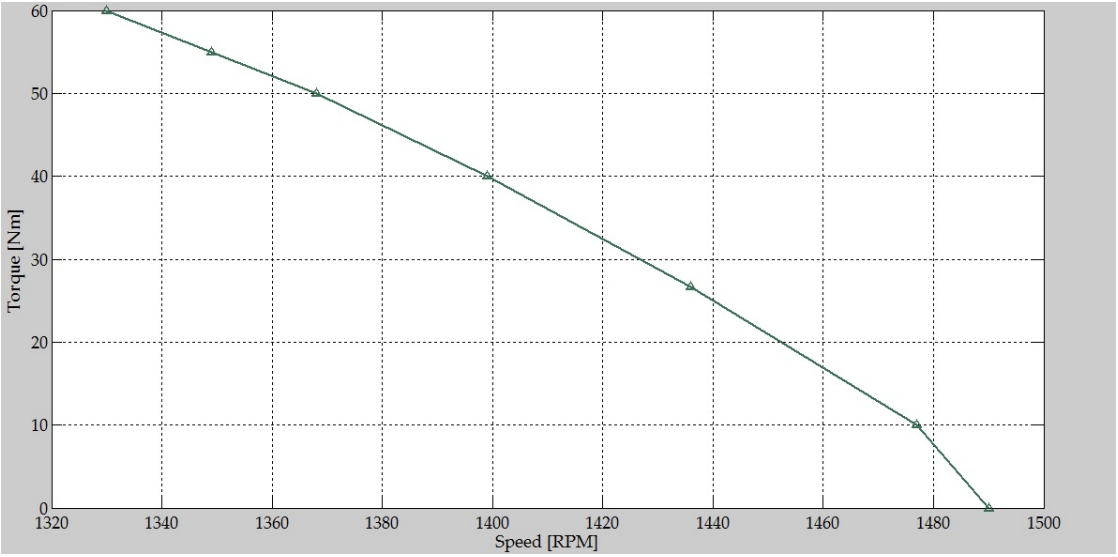


Figure 5.3: Speed-Torque characteristic

afterwards. Other intermediate intersections between the two paces do not happen, in contrast with the previous model as shown in figure 4.3.

5.3 Speed-Torque characteristic

A greater maximum torque T_M is reached (table 5.1b) thanks to a higher V_{line} which has been set at $400V$ instead of the previous $220V$. Various running has been processed and the characteristic in figure 5.3 obtained.

5.3.1 Performance

Running a simulation with the maximum torque of $60Nm$ gives a rotor speed equal to $1330RPM$ and a mechanical output power as great as

$$P_{mech} = T_M \omega_M = T_M \frac{2\pi n_M}{60} = 60 \frac{2\pi 1330}{60} = 8356.63W = 8.356kW \tag{5.3.1}$$

meanwhile the phase current is

$$\hat{I}_{ph} = 23A \qquad \rightarrow \qquad I_{ph} = \frac{\hat{I}_{ph}}{\sqrt{2}} = \frac{23}{\sqrt{2}} = 16.26A \tag{5.3.2}$$

whereas the input active power in the model (thanks to the 3-phase programmable source) is

$$P_{3ph} = 3V_{ph}I_{ph}\cos(\varphi) = 3 \cdot 230 \cdot 16.26\cos(26^\circ) = 10083W = 10.083kW \tag{5.3.3}$$

as a result the performance is

$$\eta = \frac{P_{mech}}{P_{3ph}} = \frac{8356.63}{10083} = 82.88\% \tag{5.3.4}$$

since the power losses are

$$P_{loss} = P_{3ph} - P_{mech} = 10083 - 8356 = 1727W = 1.727kW \tag{5.3.5}$$

that corresponds to the sum of $P_{R_{stat}}$ and $P_{R_{rot}}$ where

$$P_{R_{stat}} = 3R_{stat}I_{ph}^2 = 3 \cdot 1.405 \cdot 16.26^2 = 1114.4W \tag{5.3.6}$$

and knowing

$$\hat{I}_{ph_{rot}} = 22.6A \qquad \rightarrow \qquad I_{ph_{rot}} = \frac{\hat{I}_{ph_{rot}}}{\sqrt{2}} = \frac{22.6}{\sqrt{2}} = 15.98A \tag{5.3.7}$$

brings to have

$$P_{R_{rot}} = 3R_{rot}I_{ph}^2 = 3 \cdot 1.395 \cdot 15.98^2 = 1068W \tag{5.3.8}$$

$$P_R = P_{R_{stat}} + P_{R_{rot}} = 1114.4 + 1068 = 2182.4W \tag{5.3.9}$$

It emerges that there is some disparity between the two allegedly same values of P_{loss} and P_R even though not so great and equal to a error ε about

$$\varepsilon = \frac{|\Delta P|}{P_{loss}} = \frac{|P_{loss} - P_R|}{P_{loss}} = \frac{2182 - 1727}{2182} \simeq 20\% \tag{5.3.10}$$

the causes of this error growth, from what we have calculated in 4.3.17 up to the value in 5.3.10 could be explained by the rising of the additional losses P_{add} related to friction, windage and stray-load losses.[1]

[1] any leakage flux that intercepts nearby conductive materials such as the motor support structure will give rise to eddy currents and be converted to heat. There are also radiative losses due to the oscillating magnetic field but these are usually small.

6 Steady state power control

This chapter will explain how the DC/AC inverter controls the delivered power thanks to sensing the DC voltage of the DC bus V_{dc}, and the power factor φ between the voltage V_{grid} and current I_{grid} succeeding to inject only real power P_{grid} and maintaining reactive power Q_{grid} equal to zero. Further controlling methods will be studied afterwards.

6.1 Classic power flow control

Although the theory assumes a three phase line grid, it is also correct to implement it in a single phase network. Assuming that the grid can be summarized by a plain inductance (figure 6.1) the mathematical steps are as follow

$$\vec{I_{grid}} = \frac{\vec{V_{inv}} - \vec{V_{grid}}}{\jmath X} \tag{6.1.1}$$

$$\vec{I_{grid}} = \frac{V_{inv}\angle\delta}{X\angle 90^\circ} - \frac{V_{grid}\angle 0^\circ}{X\angle 90^\circ} \tag{6.1.2}$$

$$\vec{I_{grid}} = \frac{V_{inv}\angle\delta - 90^\circ}{X} - \frac{V_{grid}\angle -90^\circ}{X} \tag{6.1.3}$$

$$\vec{I_{grid}}^{\,*} = \frac{V_{inv}\angle 90^\circ - \delta}{X} - \frac{V_{grid}\angle 90^\circ}{X} \tag{6.1.4}$$

so the overall power is

$$S_{grid} = \vec{V_{grid}} \cdot \vec{I_{grid}}^{\,*} = V_{grid} \cdot \vec{I_{grid}}^{\,*} = \frac{V_{grid} \cdot V_{inv}\angle 90^\circ - \delta}{X} - \frac{V_{grid}^2\angle 90^\circ}{X} \tag{6.1.5}$$

$$P_{grid} = \Re(S) = \frac{V_{grid} \cdot V_{inv}}{X}\cos(90^\circ - \delta) = \frac{V_{grid} \cdot V_{inv}}{X}\sin(\delta) \tag{6.1.6}$$

It is drawn to conclusion that with phase dislocation δ we are able to control the delivered active power P_{grid} (figure 6.3b). Thanks to the trigonometric function sin a linear relationship between φ and P_{grid} can be exploited up to 30° (figure 6.2). As for reactive power the condition of $Q_{grid} = 0$ is reached when

$$Q_{grid} = \Im(S) = \frac{V_{grid} \cdot V_{inv}}{X}\sin(90^\circ - \delta) - \frac{V_{grid}^2}{X} = \frac{V_{grid} \cdot V_{inv}}{X}\cos(\delta) - \frac{V_{grid}^2}{X} = 0 \tag{6.1.7}$$

$$\frac{V_{grid} \cdot V_{inv}}{X}\cos(\delta) = \frac{V_{grid}^2}{X} \qquad \Longrightarrow \qquad V_{inv}\cos(\delta) = V_{grid} \tag{6.1.8}$$

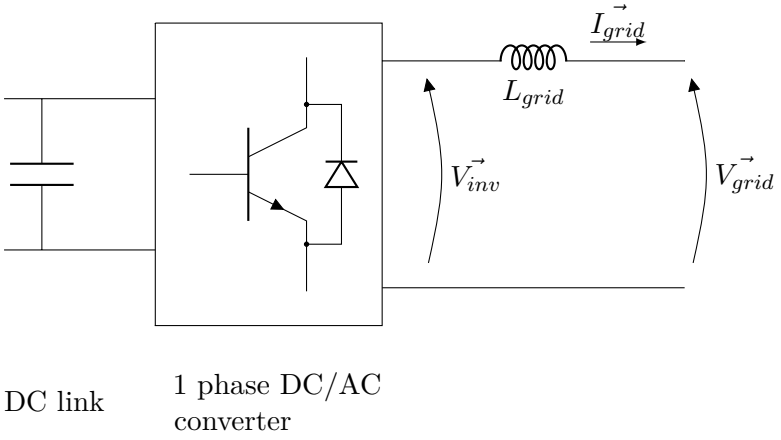


Figure 6.1: Representation of single phase grid passing through converter

As a consequence we have to control the amplitude of the modulation index m_a in order to vary V_{inv} (figure 6.3c) since the angle δ is associated with the real power P_{grid}. A phasor representation is shown in figure 6.3. An example of this control could be represented when the DC voltage V_{dc} increases in value because of additional real power coming from the renewable source (V_{inv} increases). Consequently the grid current I_{grid} grows changing its phase with V_{grid} (from figure 6.3d to figure 6.3b). At this point the inverter control must adjust δ moving to δ' (figure 6.3b) as well as the modulation index m_a in order to work with $\cos\varphi = 1$ condition (figure 6.3c) ending up to figure 6.3d where $V'_{inv} > V_{inv}$ along with $\delta' > \delta$. Mutually, the opposite pattern is done in case of a mechanical input power decrease. The overall control relies on 2 closed loops for the separate regulation of δ and m_a governed by conventional PI regulators:

LOOP 1 *capacitor voltage error* (with respect to a reference value) used to generate the angle δ between the inverter measurement V_{inv} and grid voltage V_{grid};

LOOP 2 *power factor error* (with respect to $\cos\varphi = 1$ or directly using $\varphi = 0$) used to generate the modulation index m_a for the control of the inverter voltage amplitude V_{inv}.

It is recommended to provide a value related to a reference condition in order to speed up the convergence for both δ and m_a, which are respectively δ_0 and m_{a0}. Furthermore, proper limits for the PI regulators has to be taken into account since δ may vary between 0° and 30° (for maintaining the linear relationship shown in figure 6.2) and m_a between 0 and 1.

6.1.1 Drawbacks

The major drawback of this control can be summarised in having a coupling between δ and m_a which does not allow to obtain two independent close loops from each other as the 6.1.6 and 6.1.7 show. In fact, V_{inv} affects the equation 6.1.6 and contemporarily plays a major role in deciding how much reactive power Q_{grid} can circulate according to 6.1.7. There is even the problem of the time constants to take into account down to the transfer function that we obtain.

6.2 Decoupling of the system

A solution can be represented by working in the *grid dq reference frame* rotating at synchronous speed of $50Hz$. Assuming a negligible R (as done previously) and watching the figure 6.3a

$$V^d_{grid} = V_{grid} \qquad V^q_{grid} = 0 \tag{6.2.1}$$

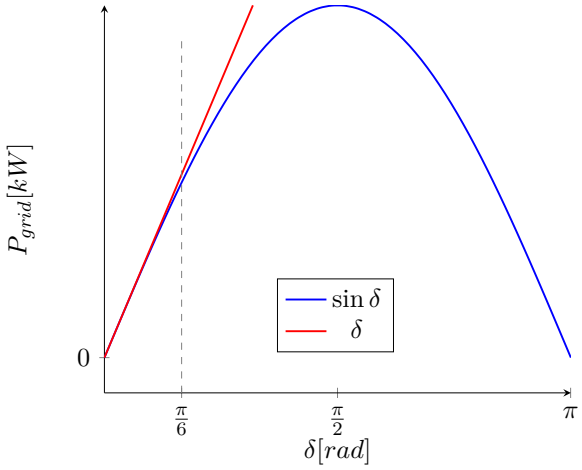


Figure 6.2: Linear relationship of $\sin\delta$

$$P_{grid} = V_{grid} I_{grid} \cos(\varphi) = V_{grid} \frac{V_{inv} \sin(\delta)}{X} = \frac{V_{grid} V_{inv}^{q}}{X} \tag{6.2.2}$$

$$Q_{grid} = V_{grid} I_{grid} \sin(\varphi) = V_{grid} \frac{V_{inv} \cos(\delta) - V_{grid}}{X} = V_{grid} \frac{V_{inv}^{d} - V_{grid}}{X} \tag{6.2.3}$$

As we can see the decoupling lets only V_{inv}^{q} commands the real power P_{grid} while Q_{grid} is entirely dependent on V_{inv}^{d}. The transformation from a stationary frame to a rotational one with pulsation equal to $\omega = 2\pi 50 = 314,15 \frac{rad}{s}$ yields to see the phasor diagram in still position instead of rotating at ω speed. In addition, since this is a mere single phase system we just need to consider the same angles used in the treatise beforehand without using the 3 phase transformation as represented in 6.2.4 derived by Clarke and Park transform

$$T_{abc \to dq0} = \begin{pmatrix} \cos(\theta_{dq}) & \cos(\theta_{dq} - \frac{2}{3}\pi) & \cos(\theta_{dq} - \frac{4}{3}\pi) \\ -\sin(\theta dq) & -\sin(\theta_{dq} - \frac{2}{3}\pi) & -\sin(\theta_{dq} - \frac{4}{3}\pi) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix} \tag{6.2.4}$$

6.2.1 Clarke transform derivation

Considering a balanced three phase system as the figure 6.4 shows, we want to obtain the real and imaginary part of it. Considering a generic vector $g(\vec{t})$ we introduce the concept of *spatial vector* as follow

$$g(\vec{t}) = \frac{2}{3}(g_a(t) + g_b(t)\underline{/120^\circ} + g_c(t)\underline{/240^\circ}) = g(t)\underline{/\gamma(t)} \tag{6.2.5}$$

in which $g_a(t), g_b(t), g_c(t)$ are projections respectively of $g_a(\vec{t}), g_b(\vec{t}), g_c(\vec{t})$ onto the real axis α (figure 6.4). As a consquence $g_\alpha(t)$ is equal to

$$\begin{aligned} g_\alpha(t) = \Re[g(\vec{t})] &= \frac{2}{3}(g_a(t) + g_b(t)\cos(120^\circ) + g_c(t)\cos(240^\circ)) \\ &= \frac{2}{3}(g_a(t) - \frac{1}{2}g_b(t) - \frac{1}{2}g_c(t)) \end{aligned} \tag{6.2.6}$$

considering a balanced system in which, by definition, the zero sequence is none

$$\frac{g_a(t) + g_b(t) + g_c(t)}{3} = 0 \to g_a(t) = -g_b(t) - g_c(t) \tag{6.2.7}$$

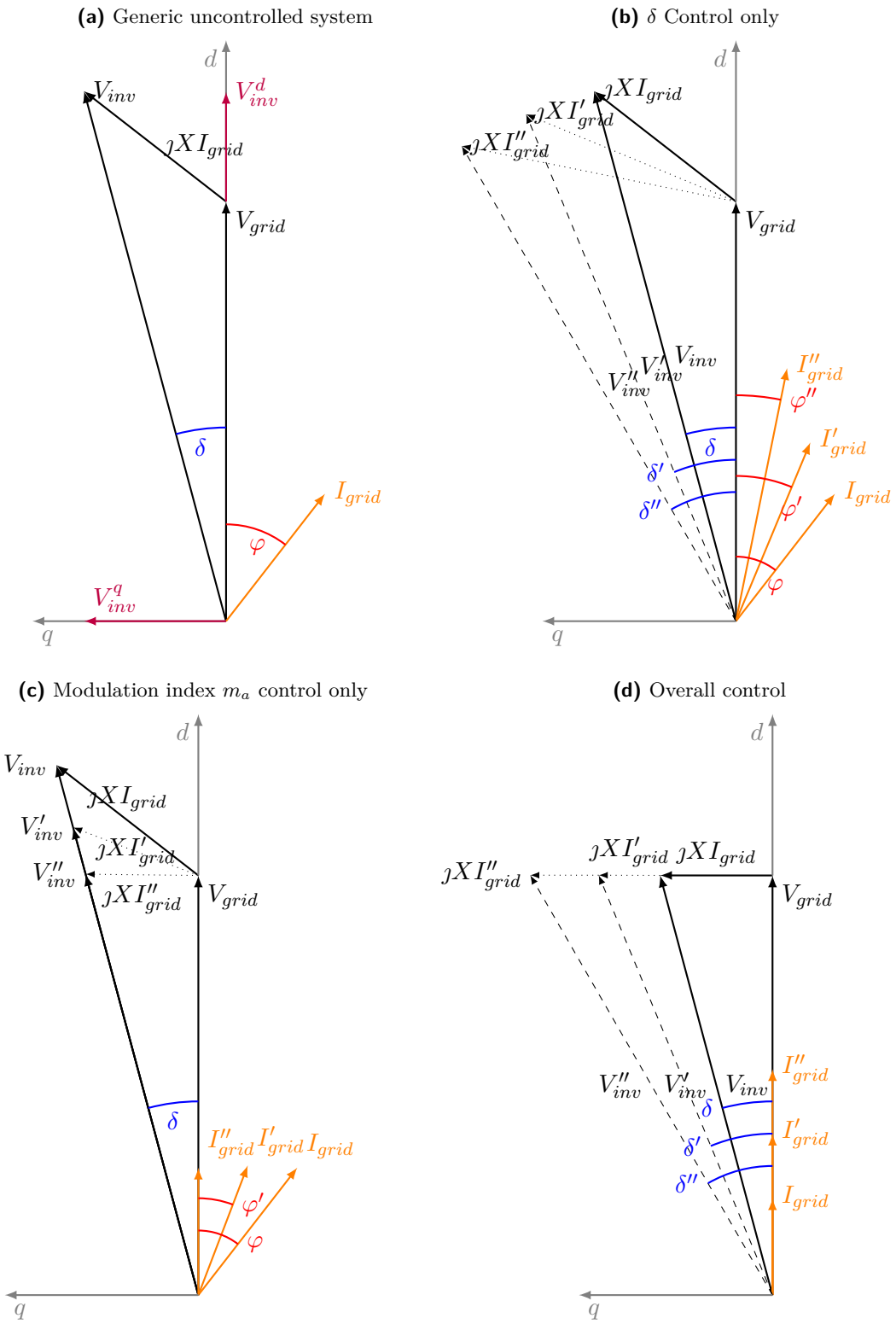


Figure 6.3: From uncontrolled to controlled system

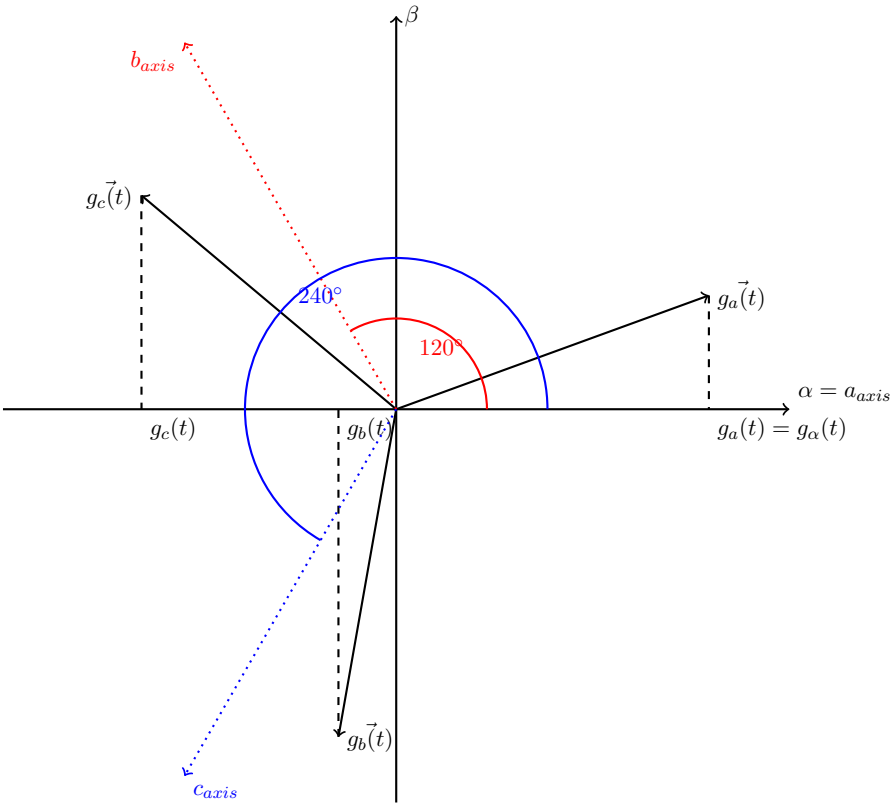


Figure 6.4: $g_a(t)$, $g_b(t)$, $g_c(t)$ From the vectors $g_a(\vec{t})$, $g_b(\vec{t})$ and $g_c(\vec{t})$

the 6.2.6 becomes

$$g_\alpha(t) = \frac{2}{3}(g_a(t) + \frac{1}{2}(-g_b(t) - g_c(t))) = \frac{2}{3}(g_a(t) + \frac{1}{2}(g_a(t))) = g_a(t) \tag{6.2.8}$$

so the real part of vector in the a_{axis} corresponds to the real part of our spatial vector as depicted in figure 6.4. Regarding the imaginary part, assuming β as the name of $\jmath$ axis

$$\begin{aligned} g_\beta(t) &= \frac{2}{3}(g_a(t) + g_b(t)\sin(120°) + g_c(t)\sin(240°)) \\ &= \frac{2}{3}(g_a(t)\sin(0°) + \frac{\sqrt{3}}{2}g_b(t) - \frac{\sqrt{3}}{2}g_c(t)) \\ &= \frac{2}{3}(0 + \frac{\sqrt{3}}{2}g_b(t) - \frac{\sqrt{3}}{2}g_c(t)) \end{aligned} \tag{6.2.9}$$

furthermore even the zero sequence has to be taken into account in order to have a complete representation of the Clarke transform as the 6.2.13 shows.

$$g_0(t) = \frac{2}{3}(\frac{1}{2}g_a(t) + \frac{1}{2}g_b(t) + \frac{1}{2}g_c(t)) \tag{6.2.10}$$

the entire transformation can be summarised by

$$\begin{pmatrix} g_\alpha(t) \\ g_\beta(t) \\ g_0(t) \end{pmatrix} = \frac{2}{3}\begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix}\begin{pmatrix} g_a(t) \\ g_b(t) \\ g_c(t) \end{pmatrix} \tag{6.2.11}$$

with a fast representation as

$$g_{\alpha\beta 0}(t) = T_{abc\to\alpha\beta 0} g_{abc}(t) \tag{6.2.12}$$

where

$$T_{abc\to\alpha\beta 0} = \frac{2}{3}\begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix} \tag{6.2.13}$$

is the Clarke matrix and

$$T_{\alpha\beta 0\to abc} = \begin{pmatrix} 1 & 0 & 1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & 1 \end{pmatrix} \tag{6.2.14}$$

is the inverse.

6.2.2 Park transform derivation

The Park transform is based on the concept of the dot product and projections of vectors onto other vectors. Assuming two vectors d and q as the axes of the new reference frame, and the well known vector $g(\vec{t})$, thanks to the graph in 6.5 we obtain

$$g\vec{_{\alpha\beta}} = g\angle\underline{\gamma_{dq} + \theta_{dq}} = g\vec{_{dq}}\angle\underline{\theta_{dq}} \quad \to \quad g\vec{_{dq}} = g\vec{_{\alpha\beta}}\angle\underline{-\theta_{dq}} \tag{6.2.15}$$

where

$$g\vec{_{\alpha\beta}} = g_\alpha + \jmath g_\beta = g\angle\underline{\gamma_{\alpha\beta}} \tag{6.2.16}$$

is the representation in the stationary frame $\alpha - \beta$. On the other hand

$$g\vec{_{dq}} = g_d + \jmath g_q = g\angle\underline{\gamma_{dq}} \tag{6.2.17}$$

is pinned as the rotational representation. It has to be highlighted that the figure 6.5 is only an istantaneous position of $d-q$ axes in respect to $\alpha - \beta$. As a consequence the dependence on time for magnitude as well as for phase can be omitted and the equations 6.2.15, 6.2.16, 6.2.17 are obvious. The 6.2.15 can also be written as a matrix relationship splitting the real and imaginary part

$$\begin{pmatrix} g_d \\ g_q \end{pmatrix} = \begin{pmatrix} \cos(\theta_{dq}) & \sin(\theta_{dq}) \\ -\sin(\theta_{dq}) & \cos(\theta_{dq}) \end{pmatrix}\begin{pmatrix} g_\alpha \\ g_\beta \end{pmatrix} = T_{\alpha\beta\to dq}\begin{pmatrix} g_\alpha \\ g_\beta \end{pmatrix} \tag{6.2.18}$$

since the determinant in $T_{\alpha\beta\to dq} = 1$ and $\cos^2(\theta_{dq}) + \sin^2(\theta_{dq}) = 1$ the inverse of 6.2.18 corresponds to its transpose yielding to

$$T^{-1}_{dq\to\alpha\beta} = \begin{pmatrix} \cos(\theta_{dq}) & -\sin(\theta_{dq}) \\ \sin(\theta_{dq}) & \cos(\theta_{dq}) \end{pmatrix} \tag{6.2.19}$$

broadening the study into a 3D system we obtain

$$T_{dq0\to\alpha\beta 0} = \begin{pmatrix} \cos(\theta_{dq}) & -\sin(\theta_{dq}) & 0 \\ \sin(\theta_{dq}) & \cos(\theta_{dq}) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{6.2.20}$$

with the inverse matrix equal to

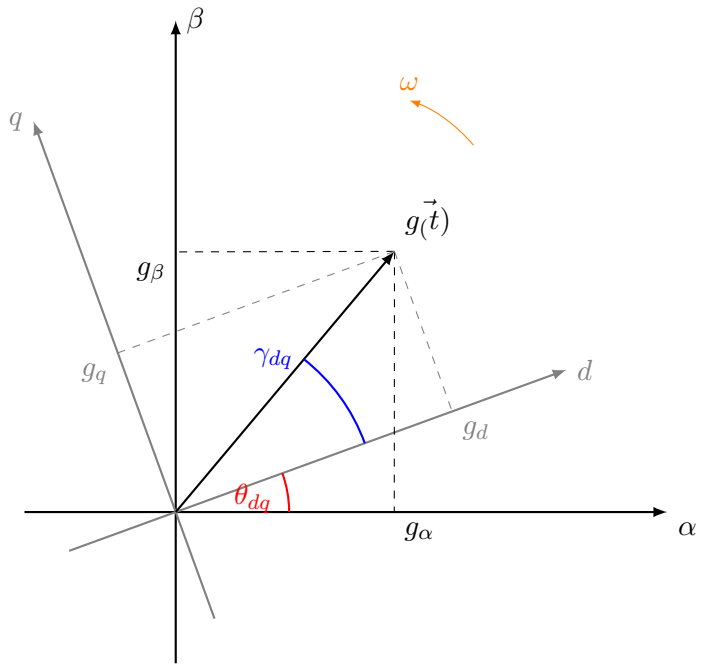


Figure 6.5: Rotational vector and different frames

$$T_{\alpha\beta 0\rightarrow dq0} = \begin{pmatrix} \cos(\theta_{dq}) & \sin(\theta_{dq}) & 0 \\ -\sin(\theta_{dq}) & \cos(\theta_{dq}) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{6.2.21}$$

finally it is clear that 6.2.4 can be calculated by the product between 6.2.13 and 6.2.21.

6.2.3 Model control

Exploiting the equations 6.2.2 and 6.2.3 we obtain the voltage references in *dq frame* since we initially have $P^*_{grid} = 4000W$ and $Q^*_{grid} = 0VAR$. Consequently:

- the reference for commanding the amount of active power delivered to the grid is equal to

$$V^{q*}_{inv} = \frac{P^*_{grid}X}{V_{grid}} = \frac{P^*_{grid}2\pi f L_{grid}}{V_{grid}} = \frac{4000 \cdot 2\pi 50 \cdot 2.15 \cdot 10^{-3}}{230} = 11.74V \tag{6.2.22}$$

- the value of the d voltage for delivering $0VAR$ to the grid is

$$V^{d*}_{inv} = V_{grid} = 230V \tag{6.2.23}$$

maintaining those two values will lead into our goal. The results of the 6.2.22 and 6.2.23 are further transformed into amplitude values whereby

$$\hat{V}^{q*}_{inv} = \sqrt{2}V^{q*}_{inv} = \sqrt{2} \cdot 11.74 = 16.6V \qquad \hat{V}^{d*}_{inv} = \sqrt{2}V^{d*}_{inv} = \sqrt{2} \cdot 230 = 325V \tag{6.2.24}$$

this has to be done because MATLAB/SIMULINK fourier block outputs are reffered to the amplitude of the measurements instead of the magnitude.

6.2.4 Drawbacks

The major issue is due to the need of a constant DC voltage source otherwise the overall system even though it properly works respecting the reference values, it would become unstable since with a decrease in V_{dc} the modulation index m_a must rise reaching the limit of 1 as fast as V_{dc} falls thus leading to a collapse. In addition, while in 6.1.6 and in 6.1.7, δ^* and m_a^* give a prompt idea about what amounts of P_{grid} and Q_{grid} are exchanged, in the dq control the quantities V_{inv}^{d*} and V_{inv}^{q*} do not mean a straightforward correlation to the power flow in the system. As a result, further mathematical computations have to provided into the associated script file in order to allow the operator to interact with the system setting simply P_{grid} and Q_{grid}.

6.3 Merged control

It is possible to mingle the advantages of the classic power flow control and the benefits of the decoupled system into a unique scheme, whose pattern is based on controlling the V_{dc} (done by the first method described) assuring to reach Q_{grid}^* promptly and without instability (dq control). It also has to be said that this layout is not affected by high grid time constants $\frac{L_{grid}}{R_{int}}$ that can appear when dealing with power systems.

6.3.1 Drawbacks

Unfortunately the decoupling is lost and a higher complexity required. In addition,the greater is the grid, the more challenging the calculations will be.

7 Building of the model

All the simulations for reaching the final complete AC/DC/AC converter control will be discussed in detail in order to give to the reader the highest level of acquaintance and the right approach for future works regarding the power flow control. Part of the simulations will be analised in the same section due to similarity and for the best comprehension possible. Over the different examinations, speed steps have been inserted for viewing how the system responded. In detail the 3 speeds during the 9 seconds of the simulation are:

1. from $0s$ to $3s \rightarrow n_1 = 1570RPM$;
2. between $3s$ and $6s \rightarrow n_2 = 1540RPM$;
3. $6s$ up to $9s \rightarrow n_3 = 1600RPM$.

7.1 Rectifying output for a DC load

The chapter starts illustrating a simple AC/DC conversion about what comes out from the SEIG (figure 7.1). As it shows, the induction generator rotates at fixed speed ($1570RPM$), so problems about speed variability are not taken into account. Consequently, the primary speed is converted to $\frac{rad}{s}$ and sent to the asynchronous generator. At 0.1 seconds a three phase breaker is on and the overall system is linked rectifying the AC voltage into a DC value delivering power to the passive load (75Ω)

$$V_{dc} \simeq 1.35 V_{line} = 1.35 \cdot 400 = 540V \tag{7.1.1}$$

consequently the power delivered to the load is (as long as we are in the first speed step)

$$P_{load} = \frac{V_{dc}^2}{R_{dc}} = \frac{540^2}{75} = 3888W \tag{7.1.2}$$

which is near to our target value about $4000W$.

7.2 Rectifying output for a DC load with AC filters

One of the problems about the circuit of the previous section is the quasi-square waveform of the current in which there are odd orders of harmonics except $3th$ and its multiples. A solution can be represented by using notch filters for the $5th$, $7th$, $11th$, $13th$ harmonic orders. For higher orders, a low pass filter ($f_{cutoff} = 1000Hz$) is sufficient. In figure 7.2 we have the improved system and the differences between the two current waveforms can be seen in figure

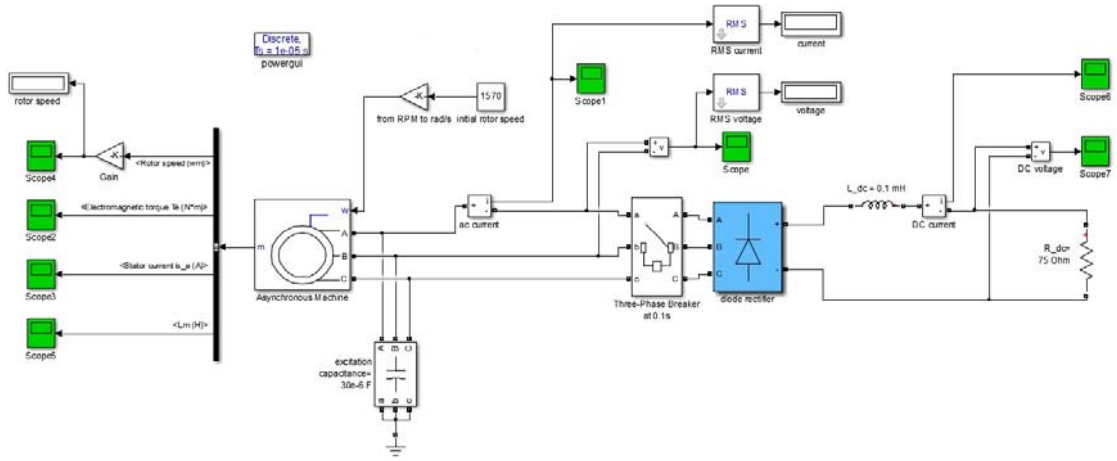


Figure 7.1: Generator-Rectifier DC load

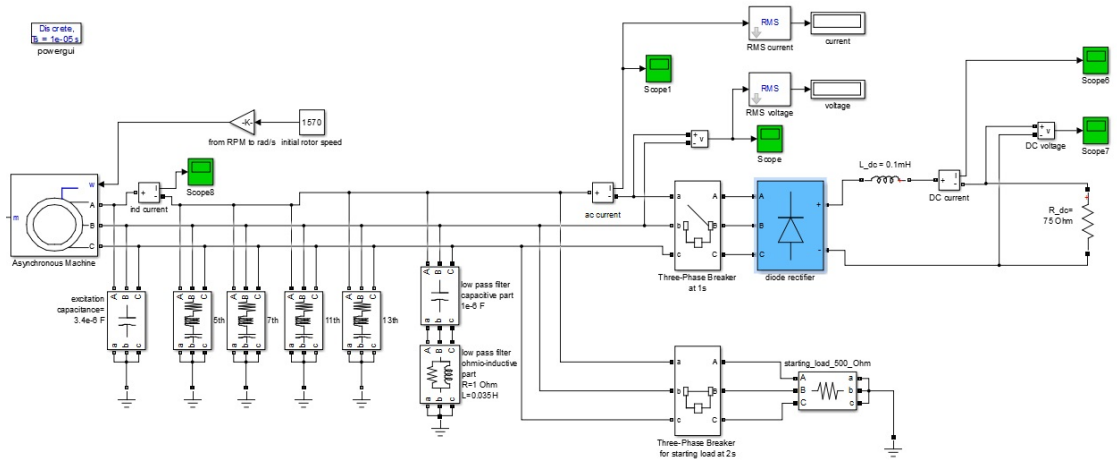


Figure 7.2: Generator-rectifier DC load with AC filters

7.3. Since less oscillation is present, the overall model benefits having a less torque ripple which yields to a more steady input power P_{mech} coming from the hydro source. Furthermore a starting load is applied to the system in order to reach the steady state values quicker than merely using only a three phase breaker: applying a starting load $R_{start} = 500\Omega$ means to have a load from the beginning and the closer P_{start} is to P_{dc}, the less the breaker transient will be. In our case

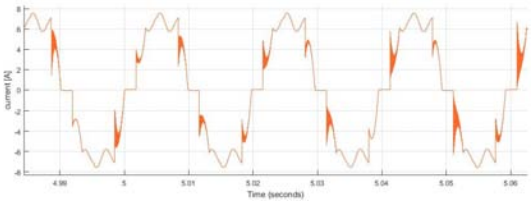
$$I_{start} = \frac{V_l}{R_{start}\sqrt{3}} = \frac{545}{500\sqrt{3}} = 0.63A \tag{7.2.1}$$

consequently the power is

$$P_{start} = 3R_{start}I_{start}^2 = 3 \cdot 500 \cdot 0.63^2 = 600W \tag{7.2.2}$$

so we are still far from having no transient when the DC load comes on. Another advantage about using filters is that we are allowed to decrease the excitation capacitance value due to the capacitors already present inside the harmonic filters which has both benefits such as cancellation of ripple and self excitation process involvement.

(a) Unfiltered current



(b) Filtered current

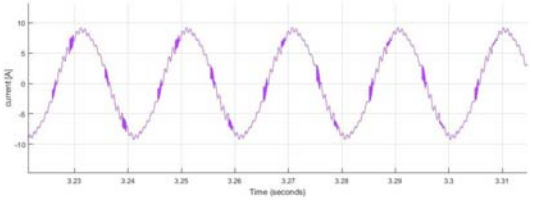


Figure 7.3: Comparison between the generator output currents

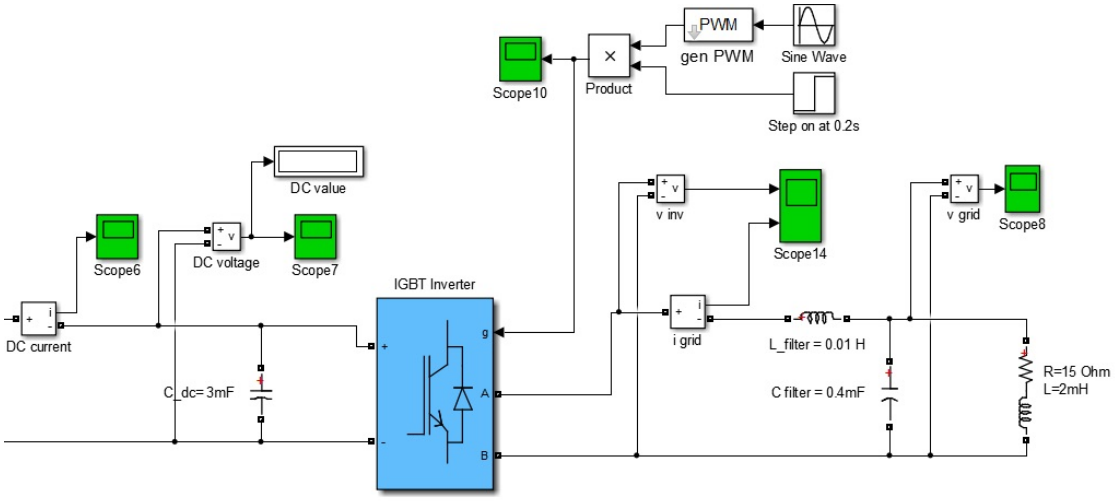


Figure 7.4: First PWM application

7.3 Open loop Pulse Width Modulation with passive AC load

The figure 7.4 shows for the first time the deployment of a PWM technique. The PWM pulses are given by a sinewave block in which we set the modulation index (between 0 and 1) as well as the pulsation ω. Afterwards, thanks to "*gen PWM*" block, we have the gate pulses for the inverter switches. Typical AC voltage and current are depicted in figure 7.5. The set values for the "*Sine wave*" are as follow

- *modulation index* m_a: for having a single phase RMS value $V_{inv} = 230V$;

$$\hat{V}_{inv} = 230\sqrt{2} = 325V \quad \rightarrow m_a = \frac{\hat{V}_{inv}}{V_{dc}} = \frac{325}{540} = 0.602 \tag{7.3.1}$$

- *pulsation* ω which is, considering a european frequence of $50Hz$;

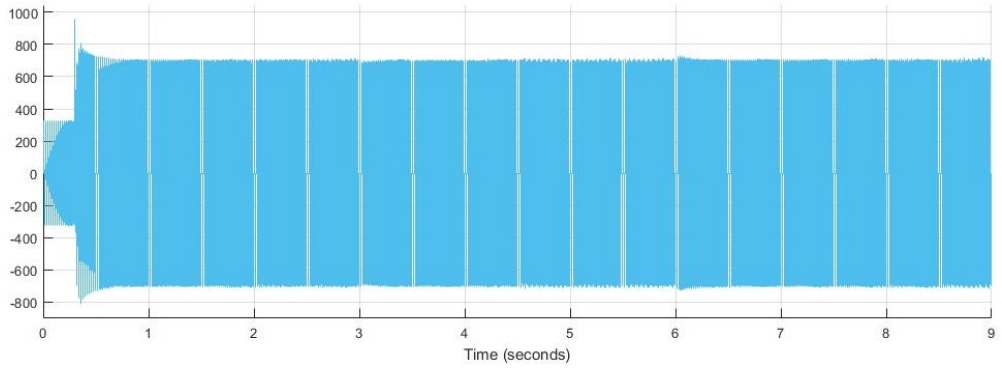
$$\omega = 2\pi 50 = 314.15\frac{rad}{s} \tag{7.3.2}$$

- the rest of the parameters are set at 0.

7.4 Open loop Pulse Width Modulation with active load

The inverter has to supply power to the single phase active grid that works as a load for the entire system. Firstly, we have to set a delivered power as great as $4kW$. Then the appropriate rate of current has to be calculated (assuming a power factor $\cos(\varphi) = 1$)

(a) PWM voltage



(b) Current shape

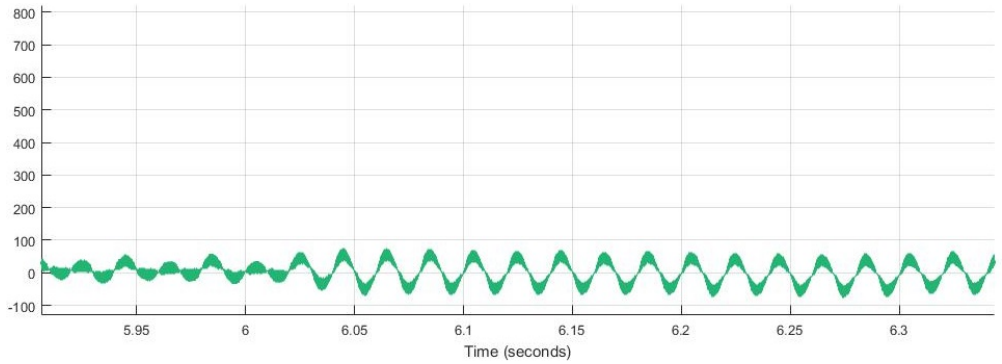


Figure 7.5: Typical PWM shapes

$$I_{grid} = \frac{P_{grid}}{\cos(\varphi)V_{grid}} = \frac{4000}{1 \cdot 230} = 17.931A \tag{7.4.1}$$

knowing that

$$L_{grid} = 2.15mH \quad \rightarrow \quad X = 2\pi 50 L_{grid} = 2\pi 50 \cdot 2.15 \cdot 10^{-3} = 0.675\Omega \tag{7.4.2}$$

the correct value of the terminal voltage that comes out from the inverter (V_{inv}) has to be equal to (assuming V_{grid} with 0 phase)

$$\vec{V_{inv}} = V_{grid}\underline{/0^\circ} + \jmath X I_{grid} = 230 + \jmath 0.675 \cdot 17.931 = 230.3\underline{/2.92^\circ} \tag{7.4.3}$$

consequently, setting the internal generation of pulses in the "*PWM generator*" block (yellow block of the 7.6) it has to be dialed:

- *modulation index*;

$$m_a = \frac{\hat{V}_{inv}}{V_{dc}} = \frac{230.3\sqrt{2}}{540} = 0.603 \tag{7.4.4}$$

  slightly higher than 7.3.1 because we have to lead V_{grid} as well as avoid to deliver Q_{grid} (figure 6.3d) ;

- *output phase* $= 2.92^\circ$.

In figure 7.5b we may observe the typical voltage shape and I_{grid} which has a typical sinusoidal waveform since we maintain a $\frac{L_{grid}}{R_{int}}$ ratio far higher than the switching period τ_s (at least 100 time as great as τ_s): indeed

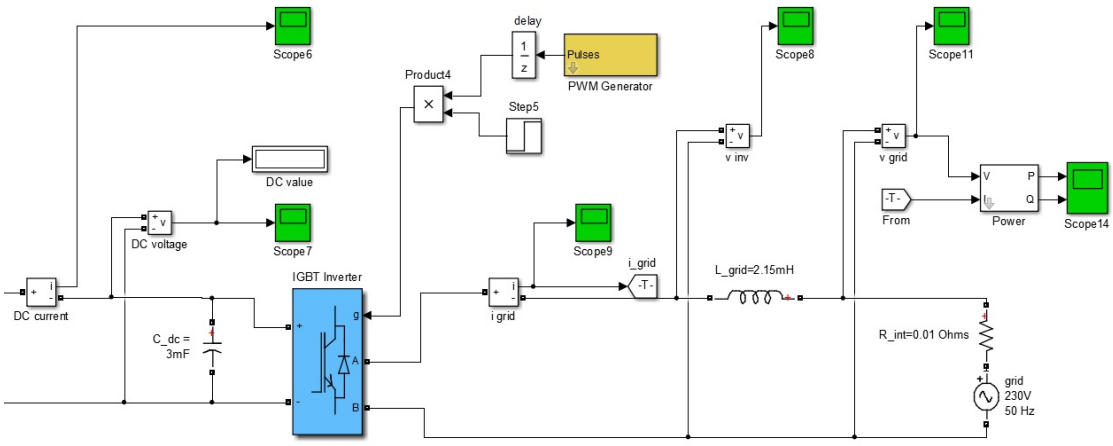


Figure 7.6: Supplying power to the single phase grid

$$\tau = \frac{L_{grid}}{R_{int}} = \frac{2.15 \cdot 10^{-3}}{0.01} = 0.215s \tag{7.4.5}$$

meanwhile the switching time interval is equal to

$$\tau_s = \frac{1}{f_s} = \frac{1}{1050} = 9.524 \cdot 10^{-4}s \tag{7.4.6}$$

in terms of frequency

$$f = \frac{1}{\tau} = \frac{1}{0.215} = 4.65Hz \qquad f_s = \frac{1}{\tau_s} = \frac{1}{9.524 \cdot 10^{-4}} = 1050Hz \tag{7.4.7}$$

if this rule is not complied we end up as in figure 7.7 where I_{grid} has several and huge harmonic content. This is due to the fact that if there is a high circuit time constant that tries to keep the current at the same value smoothening both the increase and the fall originated by the states changes in the inverter gates (hence presence R_{int} only), we will have the istantaneous value of current

$$I_{grid}^{ist} = \frac{V_{inv} - V_{grid}}{R_{int}} \tag{7.4.8}$$

every time we change the switchings state. We then are in an unstable region because the modification of the current by the source is unaffected with such a high τ. Furthermore, since there is an active load that gives a certain level of voltage, the excitation capacitors are not needed anymore.

7.5 Open loop Pulse Width Modulation rectifier

The "*PWM generator*" block has been employed even in the rectifying process as shown in figure 7.8. There is no need to set any particular phase but it is essential to digit the correct modulation index m_a^{rect} which is different from the one adopted in the single phase grid because we have to deal with a three phase system in this part of the model. As a three phase half bridge IGBT rectifier, the correct m_a^{rect} required is

$$\hat{V}_{3ph} = \frac{V_{dc}}{2} m_a^{rect} \qquad m_a^{rect} = \frac{2\hat{V}_{3ph}}{V_{dc}} = \frac{2 \cdot 230\sqrt{2}}{700} = 0.929 \tag{7.5.1}$$

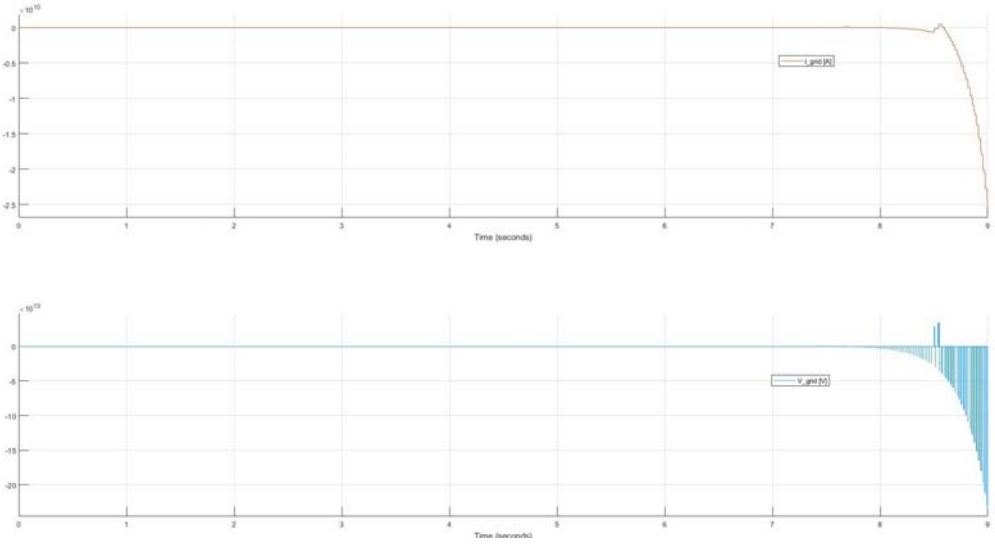


Figure 7.7: Unstable I_{grid} and V_{grid} values

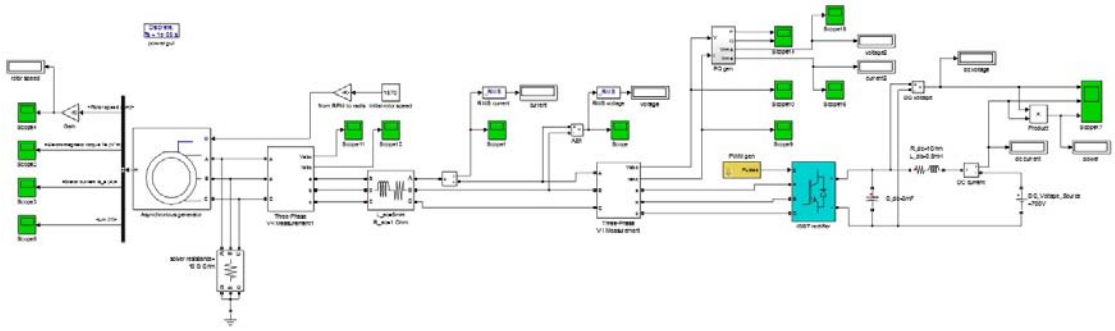


Figure 7.8: Open loop half bridge rectifier

as we can see from 7.5.1 the value related to V_{dc} has been modified from $540V$ up to $700V$ due to higher peak value $\hat{V}_{3ph}$ needs, indeed if $540V$ was the V_{dc} taken into account, the maximum $\hat{V}_{3p}$ would be

$$\hat{V}_{3ph} = \frac{V_{dc}}{2} m_{a_{max}}^{rect} = \frac{540}{2} 1 = 270V \qquad V_{3ph} = \frac{\hat{V}_{3ph}}{\sqrt{2}} = \frac{270}{\sqrt{2}} = 190.918V \tag{7.5.2}$$

which is not the desired phase voltage. As depicted in figure 7.8 the DC side is emulated thanks to a DC battery that has the same voltage of the DC capacitor related with the final model. This rectifier will be used from this section onward except on the final stage where it will be substituted with a close loop controlled rectifier as it will be undertaken for the IGBT inverter in the next section.

7.6 Close loop Pulse Width Modulation single phase IGBT inverter

This section takes into account several ways that has been ventured to control the power flow between the DC link and the single phase grid. A theoretical explanation and a detailed study in regards to how to control P_{grid} and Q_{grid} sensing V_{dc} and $\cos(\varphi)$ as well as V_d and V_q has been done in Chapter 6. Here, we will draw our attention into the issues encountered over the several trials carried out for reaching a reliable structure. We will also change the rotor speed

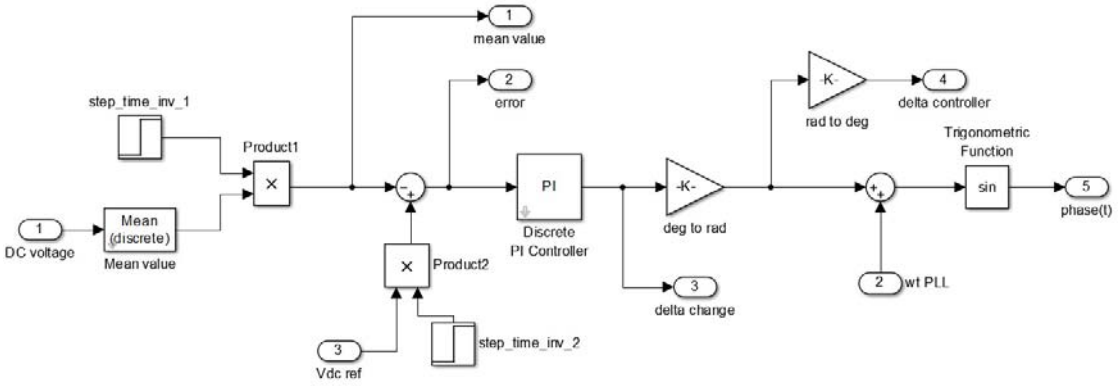


Figure 7.9: Inside the *active power control*

during the simulation in order to see whether the control reacts properly and if it does so, how long it will take to get the model back into steady state operation. For convenience reasons, only speed mechanical input models has been treated even though it is certainly possible to implement a close loop control with the torque as mechanical input of the SEIG.

7.6.1 V_{dc} Close loop

As first step, only the voltage of the DC link is controlled thanks to a feedback loop, leaving the other half with an imposed default value of m_{a0} generally set to give $Q_{grid} = 0$. The rate of V_{dc} is measured directly from the DC bus and compared with a reference rate. Having obtained the error by $V_{dc}^* - V_{dc}$, we use a PI controller to change the phase between V_{inv} and V_{grid}. The higher the error is, the quicker δ will vary. Afterwards it follows a conversion of the δ value obtained, from *degrees* to *radians*. At the end $\delta(rad)$ is finally associated with a phase locked loop (PLL) coming from V_{grid}. The main difficult is related to the PI controller. First of all, the limits have to be set by complying the constraints that theory impose, such as the possibility to have only $\delta \geq 0$ as well as the equivalence between the function $\sin\delta$ and the angle δ itself valid up to 30° (figure 6.2). In addition, a default δ_0 has been set equal to 2.92°. Therefore we impose (swap signs due to substraction algebraic sum):

- PI voltage control lower limit $-20°$ (for keeping margin to $-30°$);
- PI voltage control upper limit 2.92°.

The next step is about deciding what are the most suitable values for the proportional gain k_p and the integral gain k_i in order to have the quickest response of the system. we can proceed considering firstly a none integral gain ($k_i = 0$). Consequently, only k_p can change. At this point we may vary k_p until we reach a constant δ output that settles between ° and 20°. When we accomplished, we can move forward to the k_i value that has to erase the previous constant error quickly. At the end of several trials we come up with

$$k_p = 0.15 \qquad k_i = 1 \tag{7.6.1}$$

An overview of the system can be seen respectively in figure 7.9 and in 7.13. In addition, many "*step time*" blocks are used for convenience reasons such as to turn on or to turn off any particular signal whenever is needed: for example they can be used for getting rid of annoying initial transients that can widely postpone the steady state operation. It also has to be said that the simulation and MATLAB/SIMULINK take more minutes for completing the run every time we move forward with our project since we modify the model to a more complicated one. The main values of the V_{dc} control are gathered in figure 7.10.

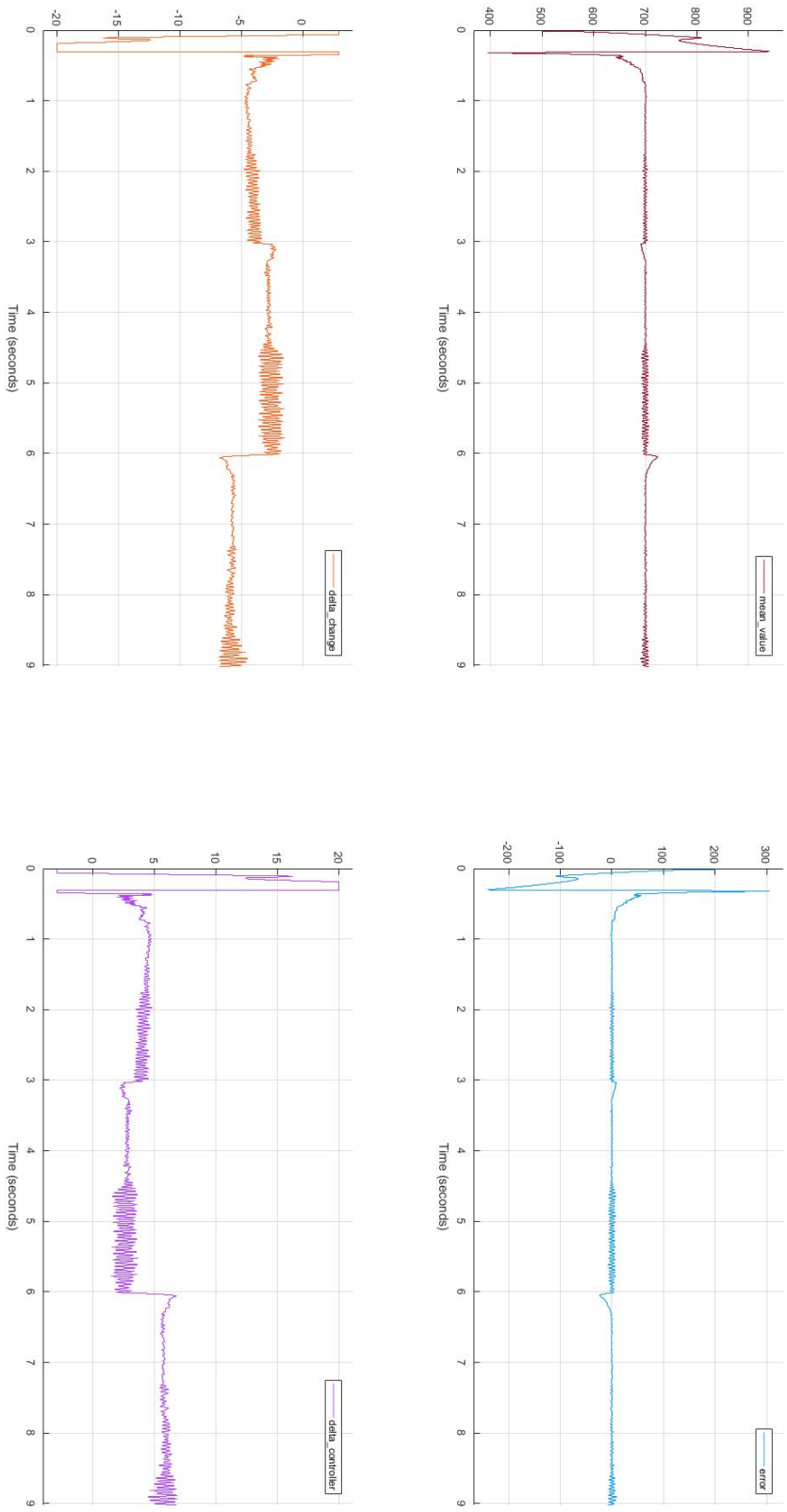


Figure 7.10: Main graphs of the active power control

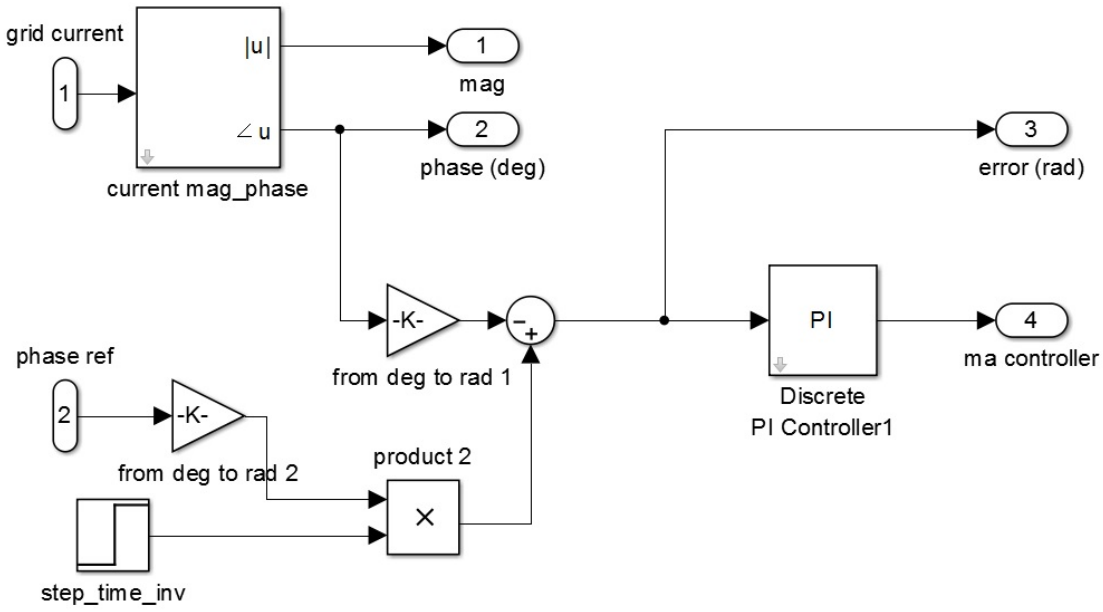


Figure 7.11: Inside the *power factor control*

7.6.2 Power factor and overall control loop

It is the further implementation of the model discussed beforehand. Now even the PF control is linked to the "*PWM generation*" block so we have a complete control as schematised in figure 7.13. It has to be said that, at the beginning, we were not able to have a stable control due to the too high $\frac{L_{grid}}{R_{int}}$ ratio. In the current model, this ratio has been lessened from $100s$ to $0.2s$ which makes everything closer to the reality of a common single phase grid. The *power factor control* is made up with the similar layout of the *active power control* except for the fact that it waives the Phase Locked Loop since its outcome is equal to m_a which should be a constant value. Furthermore, the m_{a0} is still on as seen in figure 7.13. A depicture of the *power factor control* is illustrated in figure 7.11. The control starts converting I_{grid} into both magnitude and phase with the help of a Fourier block (left hand corner of 7.11). After this passage the phase is converted into rad as well as compared with the reference value of $0rad$. Consequently the error is sent into a PI controller that gives us the m_a controller. At the end, as shown in figure 7.13, a further sum is carried out having set the default m_{a0} to 0.5. The results of this loop are gathered in figure 7.12. If we analise the PI controller we can refer to the procedure done in the *active power control* for the correct proportional and integral gains. However, we have to consider different limits for the controller since there are $m_{a0} = 0.5$. Bearing in mind that we have opposite signs in the algebraic sum nod the PI parameters are

- power factor control lower limit -0.5;
- power factor control upper limit 0.5;
- proportional gain $k_p = 0.005$;
- integral gain $k_i = 2$.

7.6.3 *dq* Control

A different approach can be carried out by implementing a decoupling of $\vec{V}_{inv}$ into direct and quadrature components. Figure 7.14 offers an overview of the overall control. The conversion from a time-dependent voltage to dq axes relies in the "*dq components*" block that is connected

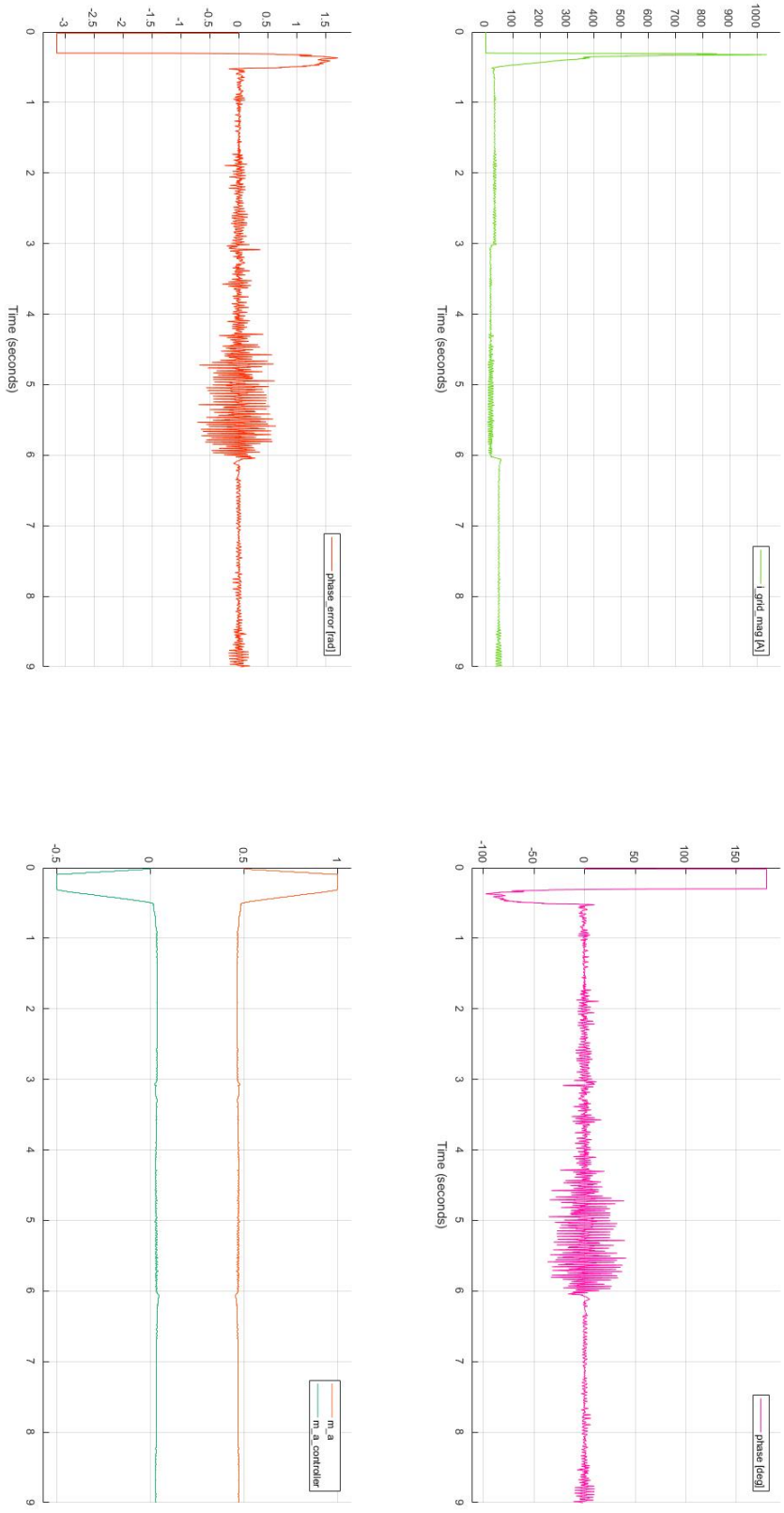


Figure 7.12: Main graphs of the phase control

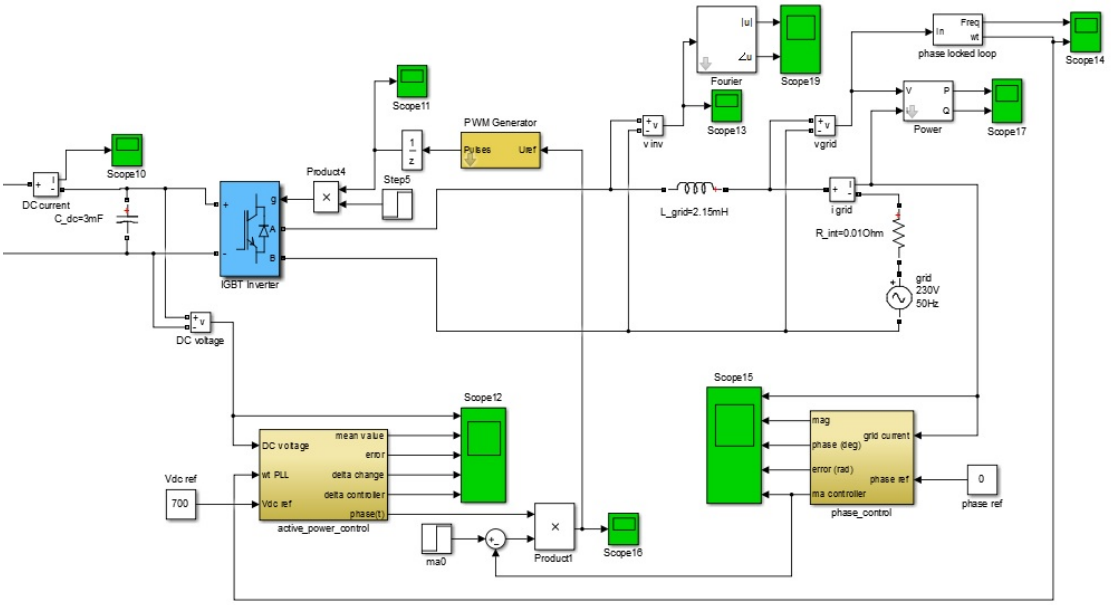


Figure 7.13: *Active power control* and *phase control* working together

to "*dq control*" through labels. On the left bottom side, the references V_{inv}^{d*} and V_{inv}^{q*} are previously calculated thanks to the passages written on the script file associated with the SIMULINK models and exposed in 6.2.24. The *dq control* block represents the close loop aimed to bring the model at the desired conditions of working. Inside the block (figure 7.15) there are two separate loops that merged at the end into one signal which will be the sinusoidal waveform sent to the "*PWM generator*" block; the layout of the feedback loop inside the subsystem is quite similar to figure 7.9 and 7.11. Nevertheless the limits and the gains are:

- *quadrature voltage axis* V_{inv}^{q} *controller*:
  - lower limit 0;
  - upper limit 20;
  - $k_p = 0.01$;
  - $k_i = 1$;
- *direct voltage axis* V_{inv}^{d} *controller*:
  - lower limit -0.5;
  - upper limit 0.5;
  - $k_p = 0.001$;
  - $k_i = 0.05$.

We can see a similarity with the PI δ and m_a limits previously adopted in figure 7.9 and 7.11. On the other hand the gains rate are fairly distant from the previous models due to quite different measurement scales. We boldly underlined the need to have a "*DC voltage source*" otherwise there is no means for controlling the power flow (section 6.2). We can see in figure 7.16 how the two components evolve during the simulation.

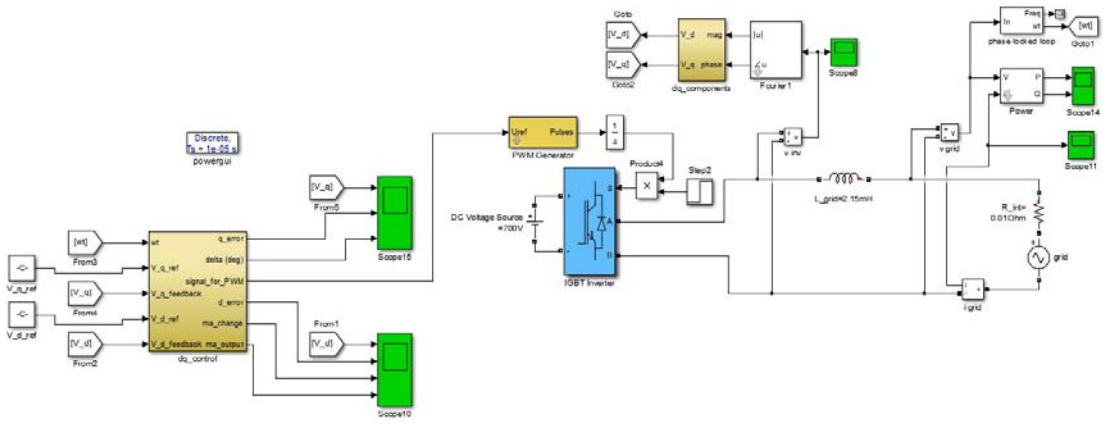


Figure 7.14: Overview of dq control approach

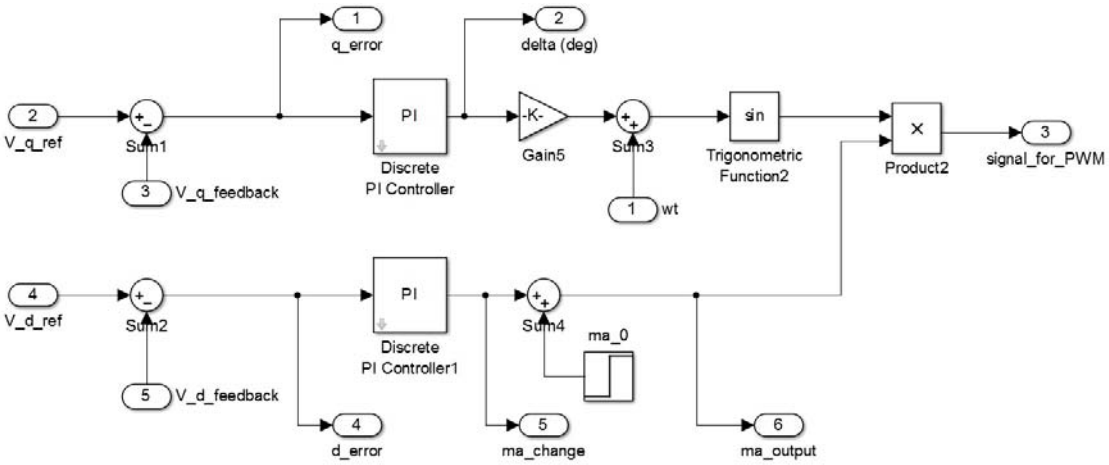


Figure 7.15: Inside the "*dq control*" block

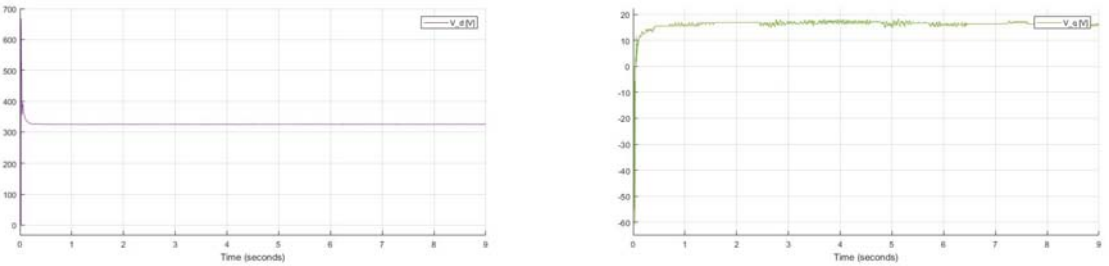


Figure 7.16: V_d and V_q equal respectively to the values calculated in the equation 6.2.24

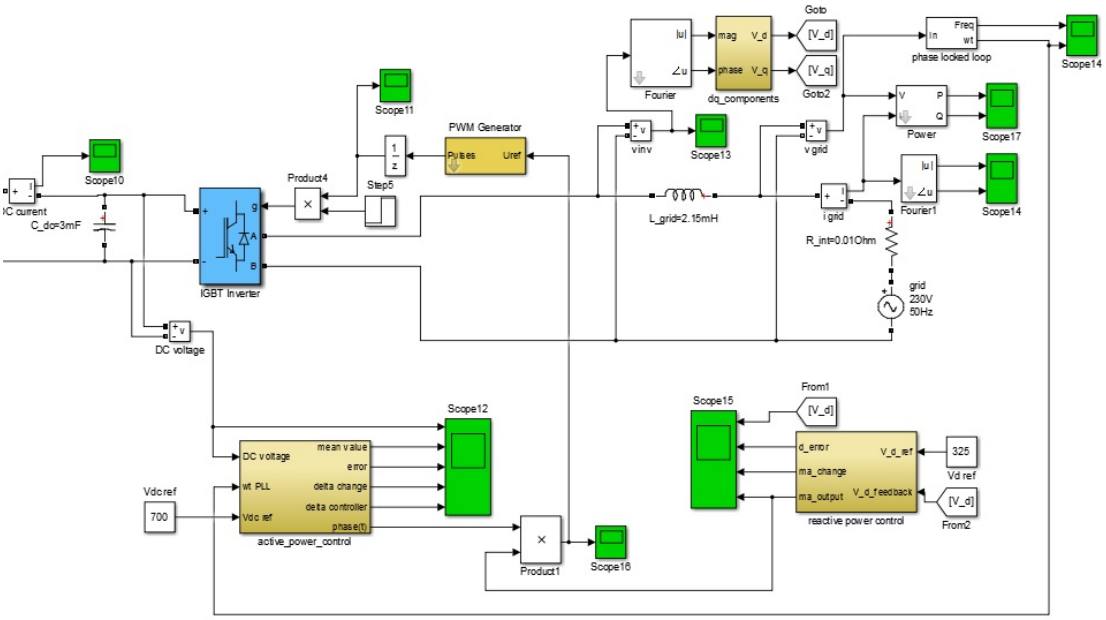


Figure 7.17: Merged control

7.6.4 Merged control

It is possible to create even a hybrid model where nothing change in relation to numerical parameters, but there is a different architecture that allows to control the Q_{grid} through V_{inv}^{d} while delivering real power P_{grid} according to the reference DC link voltage V_{dc}. It is also feasible to have the diametrically opposite reference values which means taking into account V_{inv}^{q} so as to adjust real power flow and in the meantime set 0° as $\cos(\varphi)^{*}$. One of the many possible examples is illustrated in figure 7.17. It can be thought that is not worth controlling the model with two different methods altogether because this might be interpreted as an oversizing. On the other hand this layout may be used when any other type of control would fail since we try to extract only the advantages from each previous approach. Indeed, controlling the V_{dc} leads to succeed in controlling everything with more accuracy and promptness than what we have in figure 7.13. A detailed depictures of both the active power control outcomes based by sensing the V_{dc} at the DC link and the V_d close loop results for retaining $0VAr$ throughout the operation are respectively given by figure 7.18 and figure 7.19.

7.7 Close loop rectifier

In order to reach a thorough control of the system we must draw the attention to the rectifying AC/DC process in which we previously imposed an open loop rectifier control thanks to the "*PWM generator block*". A DC voltage source simulates the V_{dc} of the DC bus used to deliver controlled power to the AC single phase side. Inductance and resistance are added so as to smooth the rectified voltage waveform as well as to obtain a clear value of the power flow. As seen in figure 7.20, the "*voltage regulator*" block corrects the amplitude value of V_{3ph} whose reference is equal to $325V$. Inside the "*voltage regulator*" we find an $abc \rightarrow dq0$ transformation where the reference $V_{3ph}^{*} = V_d$. Having calculated the error, this goes through a PI controller in which the gains are

$$k_{pvr_{rec}} = 0.0001 \qquad k_{ivr_{rec}} = 8 \tag{7.7.1}$$

the modulation index m_a^{rect} is also calculated and scoped in figure 7.22 where there is a

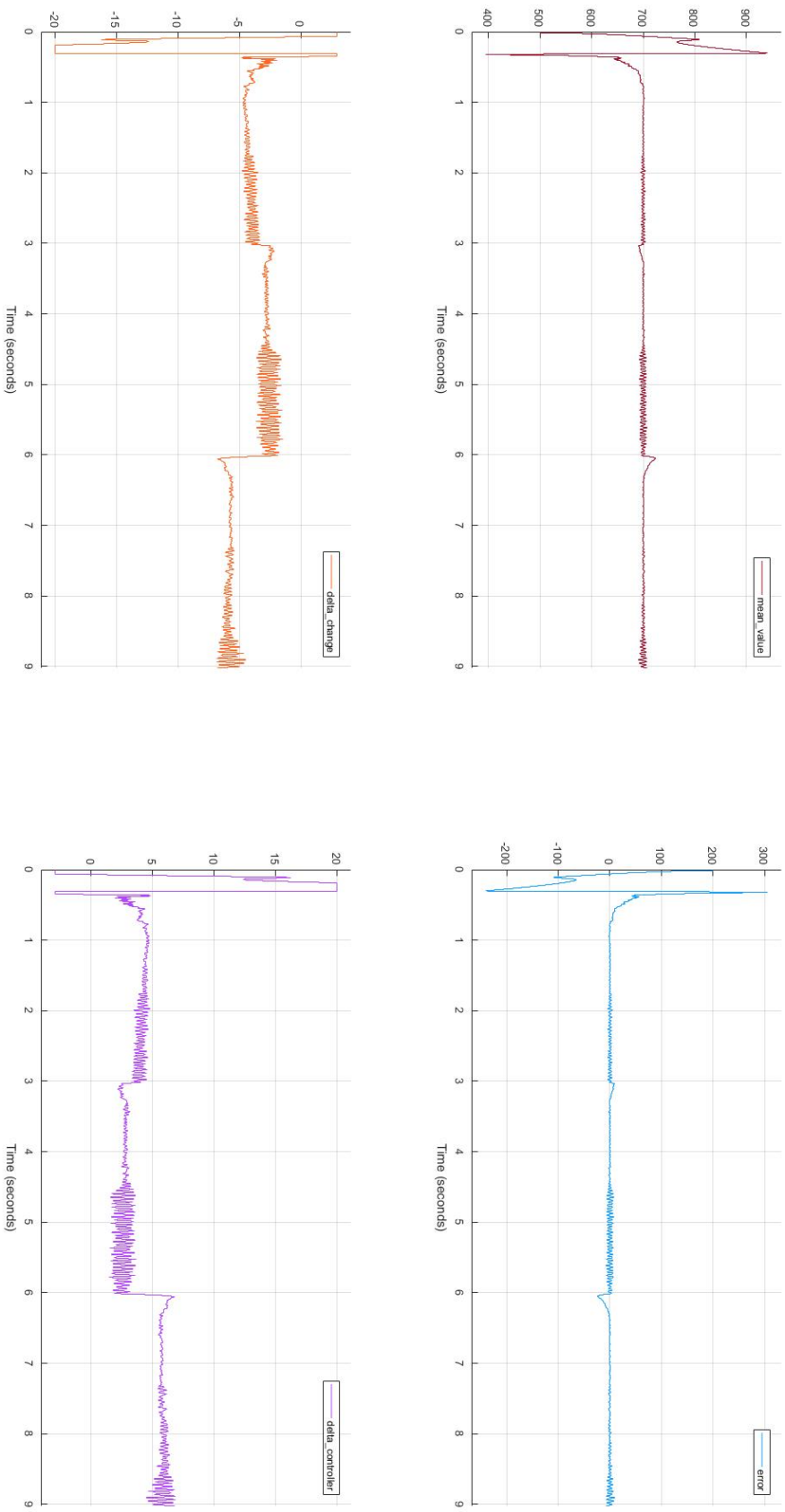


Figure 7.18: Active power control made by V_{dc} detection

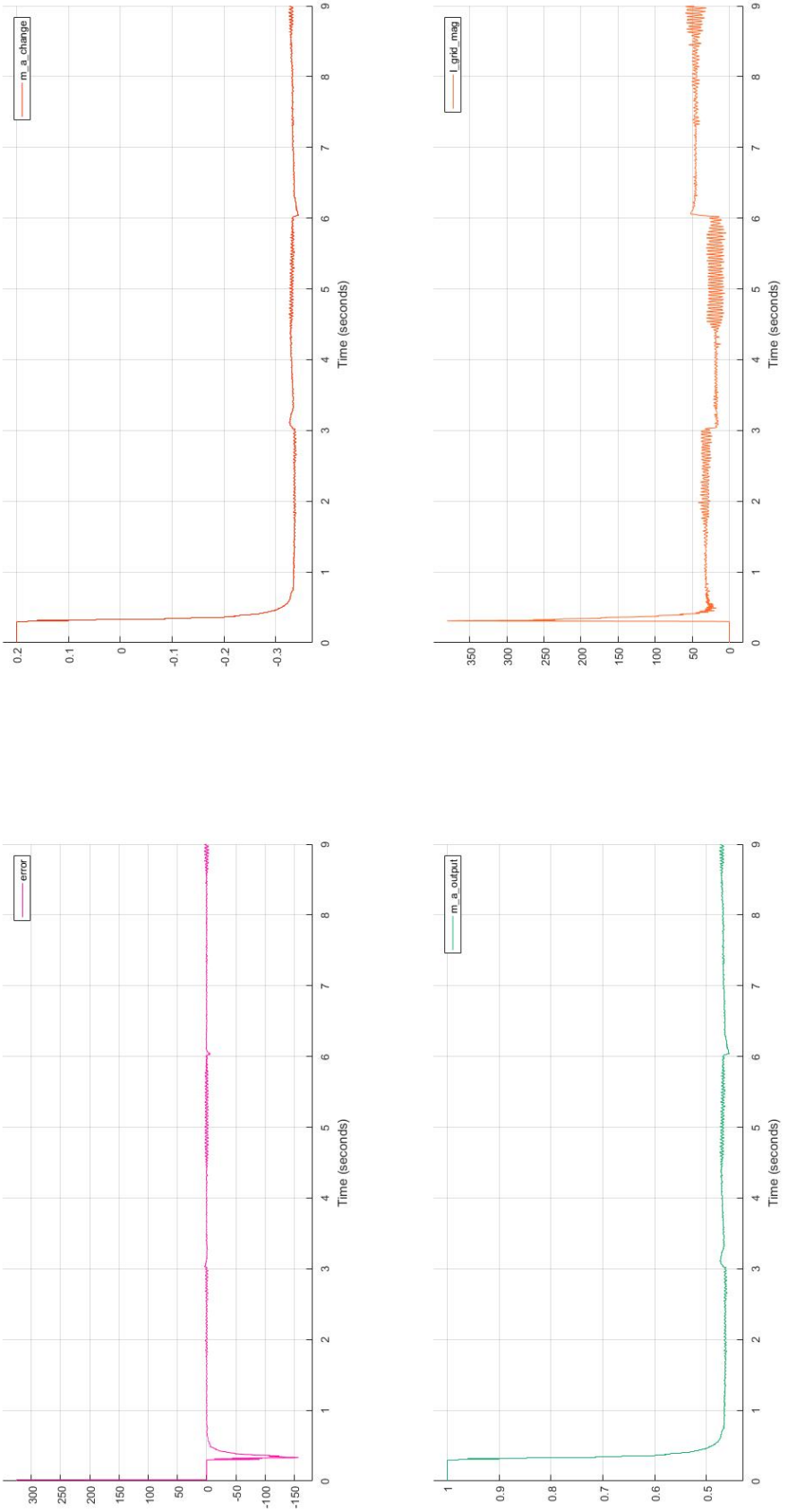


Figure 7.19: Reactive power control thanks to V_d

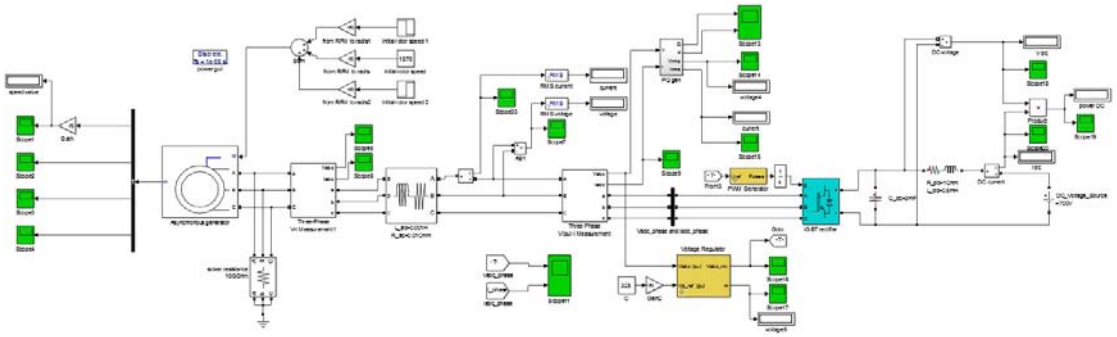


Figure 7.20: Close loop rectifier

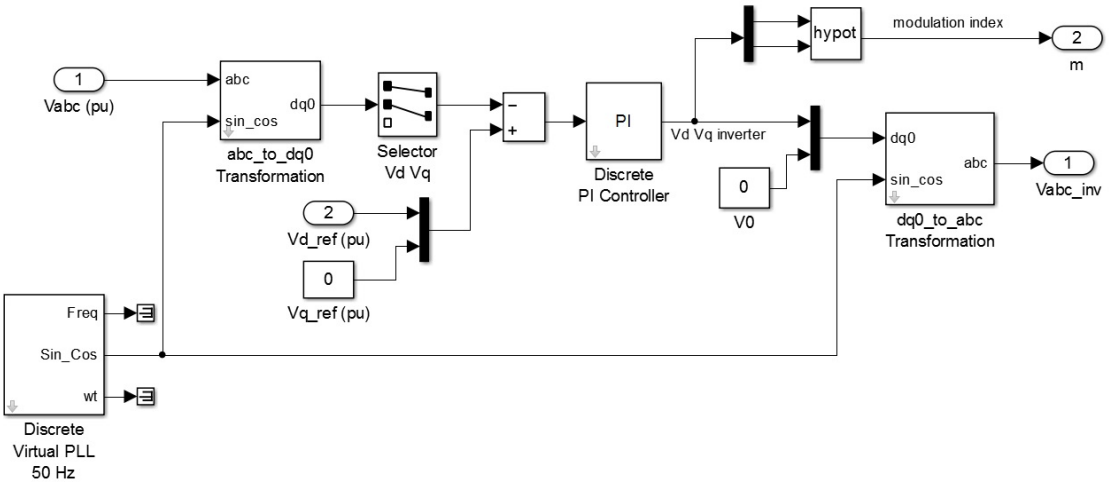


Figure 7.21: Inside the "*voltage regulator*"

correspondence with what has been theoretically calculated which is

$$m_a^{rect} = 2\frac{V_{3ph}}{V_{dc}} = 2\frac{325}{700} = 0.928 \tag{7.7.2}$$

Finally the V_d output turns back into an *abc* system again (figure 7.21) and sent to the "*PWM generator*". All the control brings to scope the figure 7.22 where the voltage regulator peaks conjuction denotes the path of the modulation index m_a^{rect}.

7.8 Complete control

The total control is achieved and illustrated in figure 7.23 where the reference voltage for the rectifier PWM is set onto the q axis instead of onto the d axis since an ac inductive impedance has been added before the rectifier. The overall power exchanged is collected in figure 7.24. As denoted the Q is mantained as close as possible to zero. Whereas the active power varies because of the mechanical power input changes.

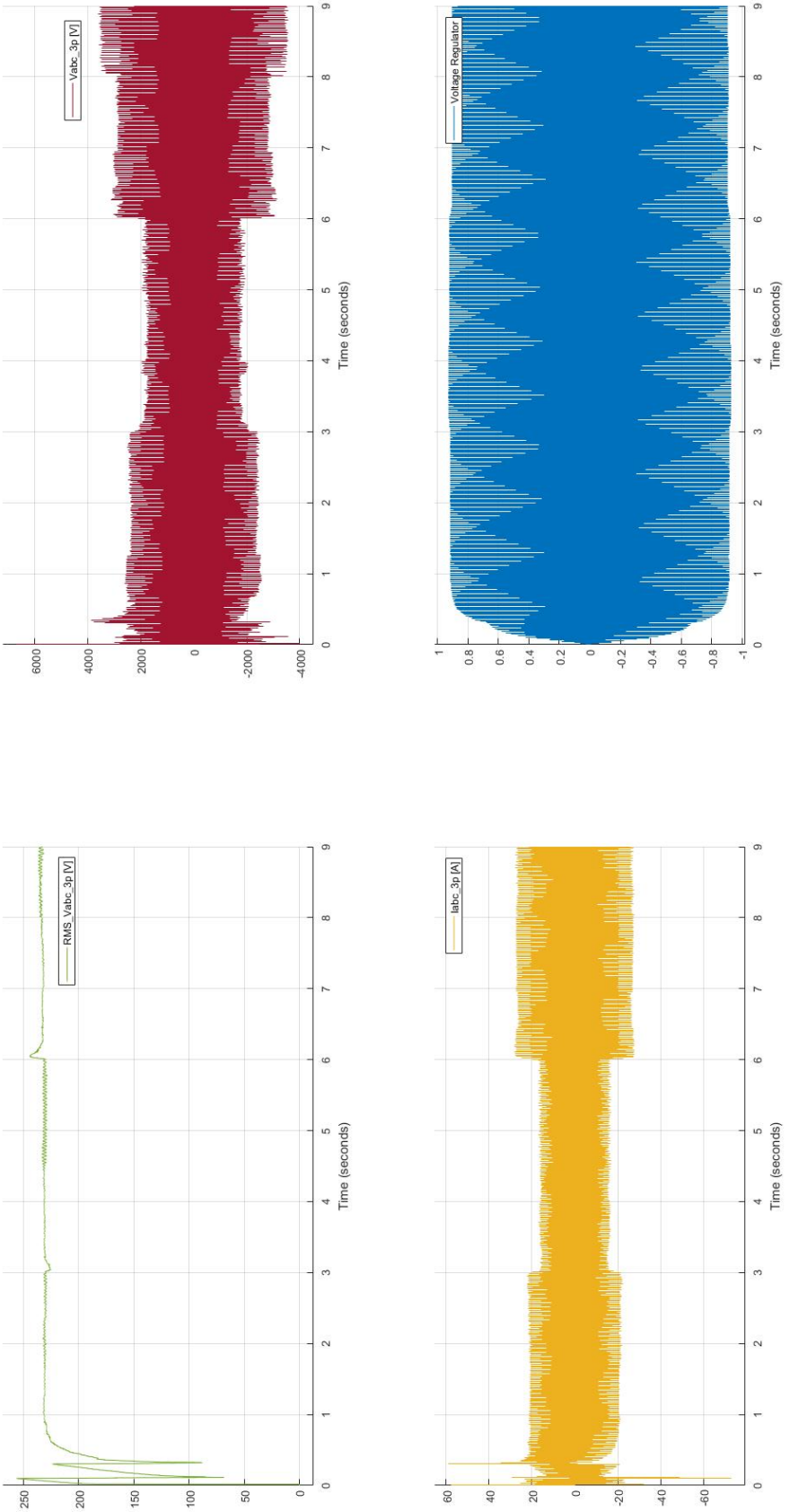


Figure 7.22: Voltage regulator on action

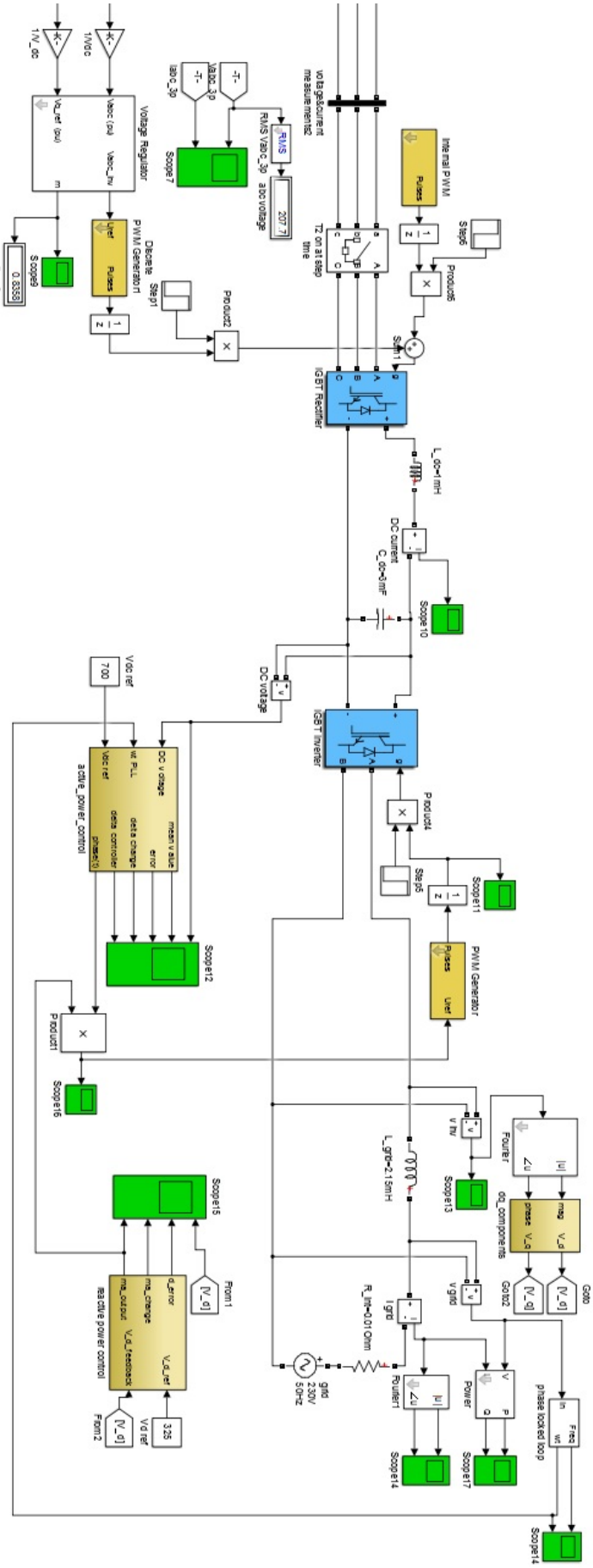


Figure 7.23: Complete control

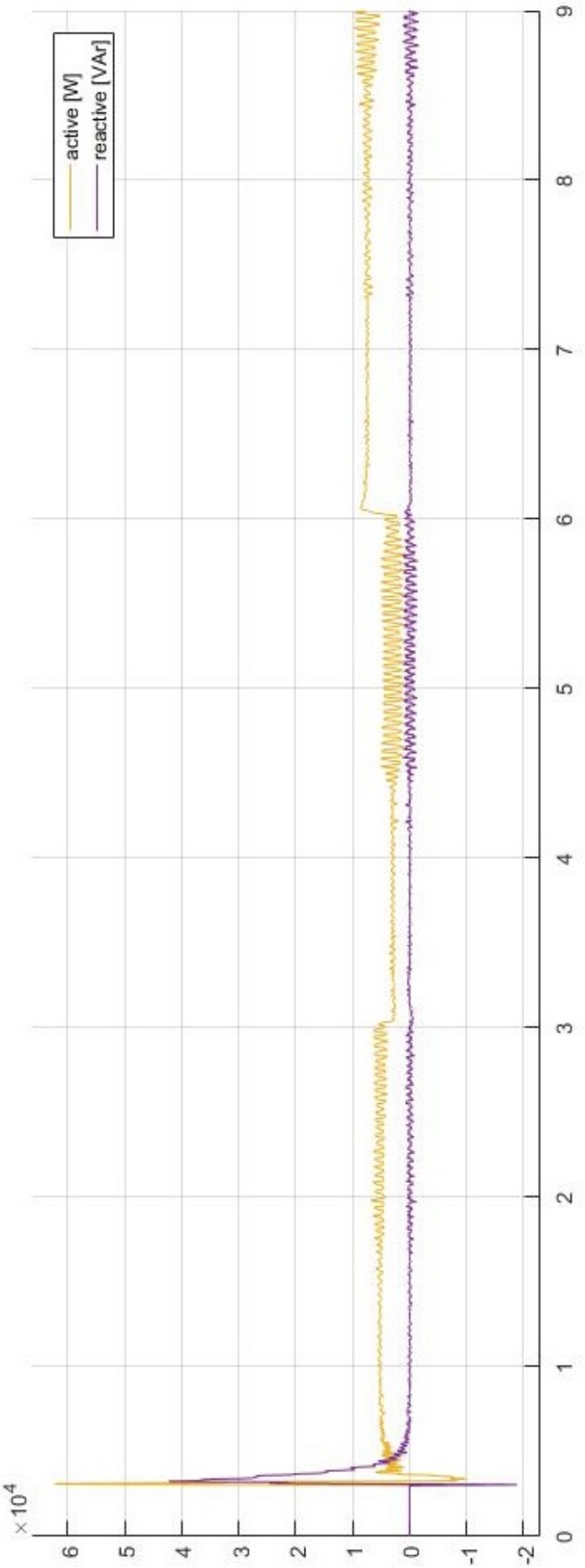


Figure 7.24: Active and reactive power in this simulations

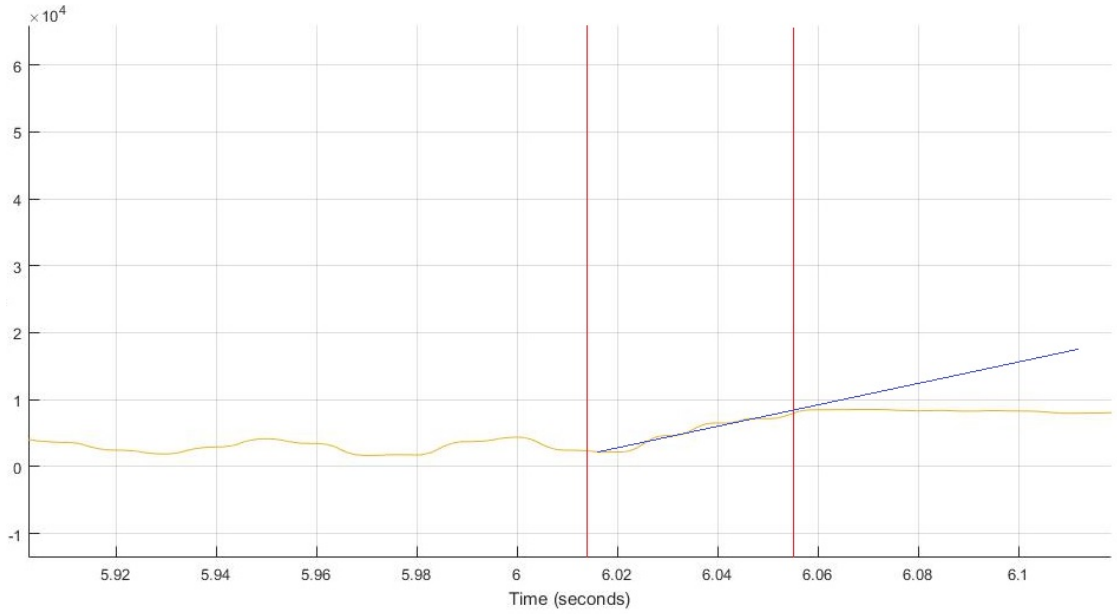


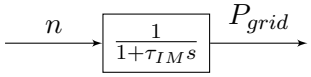
Figure 7.25: zoom of the active power trend in the point where the speed step is turned on

7.9 Transfer function

The transfer function is easy to calculate and it can be found measuring the derivative of the active power P_{grid} when the speed step is turned on. Let is take for example the change in speed happened at $6s$ moving from $1540RPM$ to $1600RPM$. Thanks to figure 7.25, the time constant for the first order block representing the induction machine is

$$\tau_{IM} = 6.052 - 6.011 = 0.041s \tag{7.9.1}$$

this is a semplified calculation taking the time when the power are steady again and substracting 6 seconds. As a consequence, the block that represents all the system is



where $\tau_{IM} = 0.041s$. Quoting what said in 7.4.5, the time constant of the grid might affect the stability of the entire system pushing the phase dislocation, with the pole in $\frac{1}{\tau}$, to a general instability. So there is a compromise in regards to the values of resistance and inductance chosen or naturally inside of the grid.

8

Programming with a dsPIC microcontroller

We will draw our attention to implement part of what has been achieved in the simulations into a physical component in order to create a control for the system exploiting what so far studied. For doing this, we need to move in another branch of the engineering that deals with programming as well as interfacing with different softwares and electronic components. Basically, we need:

1. a language of programming for setting the desired commands. We will use *C++*;
2. a microCHIP that acquires all the information written in $C++$ and store them into a memory. In detail our piece will be a *dsPIC30F4011*;
3. an intermediate part that connects computer and dsPIC30F4011 namely a *breadboard* and *PICkit 2*.

8.1 Programming in *C*

C is a general-purpose programming language with features economy of expression, modern flow control and data structures, and a rich set of operators. C is not a very high level language, nor a big one, and it is not specialized to any particular area of application. However, its absence of restrictions and its generality makes it more convenient and effective for many tasks than supposedly more powerful languages. For a complete study of this language the reader is exhorted to read [5] of the bibliography.

8.2 dsPIC30F4011

The microcontroller that we use belongs to the PIC range, which is produced by Microchip Inc. (see http://www.microchip.com/). This range of microcontrollers contains hundreds of different chips, grouped into several product families. Microcontrollers with features similar to those in the PIC range are available from many other manufacturers (e.g. Texas Instruments, Intel, Atmel, etc.), but to keep things simple in this module we always use the same microcontroller. Like all chips in the 30F family, the dsPIC30F4011 is a 16-bit microcontroller (figure 8.1a). What that means is that each of its registers (memory locations) stores one 16-bit binary number. The dsPIC chips we use come in a 40-pin dual in-line package (DIP). In this context, the word "package" simply refers to the actual body of the chip which is the block of dark plastic where the tiny silicon integrated circuit is permanently encased (rather than the wrapping the chip is sold in). DIP chips have a standard pin size and spacing that makes them

suitable for plugging straight into a breadboard. It is possible to buy the exact same dsPIC in a much smaller surface-mount package, but it cannot then be used easily in a breadboard. The DIP package is therefore much more convenient for the experimental work we do.

8.3 Breadboard

A breadboard is a construction base for prototyping of electronics (figure 8.1b). Because the solderless breadboard does not require soldering, it is reusable. This makes it easy to use for creating temporary prototypes and experimenting with circuit design. Breadboards are available from several different manufacturers, but most share a similar layout. The layout of a typical solderless breadboard is made up from two types of areas, called strips. Strips consist of interconnected electrical terminals.

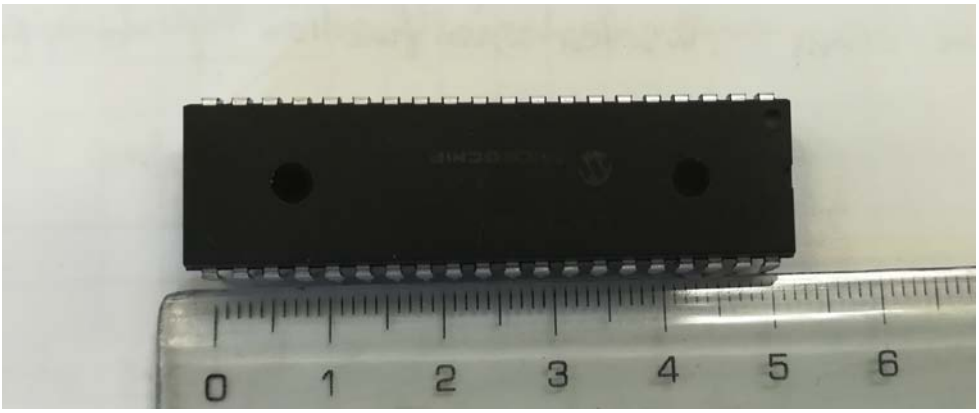
8.3.1 Terminal strips

These are the main areas, to hold most of the electronic components. In the middle of a terminal strip of a breadboard, one typically finds a *notch* running in parallel to the long side. The notch is to mark the centerline of the terminal strip and provides limited airflow (cooling) to the strips. The clips on the right and left of the notch are each connected in a radial way; in our case, 6 clips in a row on each side of the notch are electrically connected. The 6 columns on the left of the notch are marked as A, B, C, D, E and F whereas the ones on the right are marked G, H, I, J, K and L. When pin package integrated circuit (such as our typical dsPIC30F4011) is plugged into a breadboard, the pins of one side of the chip are supposed to go into column E while the pins of the other side go into column I on the other side of the notch. The rows are numbered 1 - 28 or whatever number of columns there are.

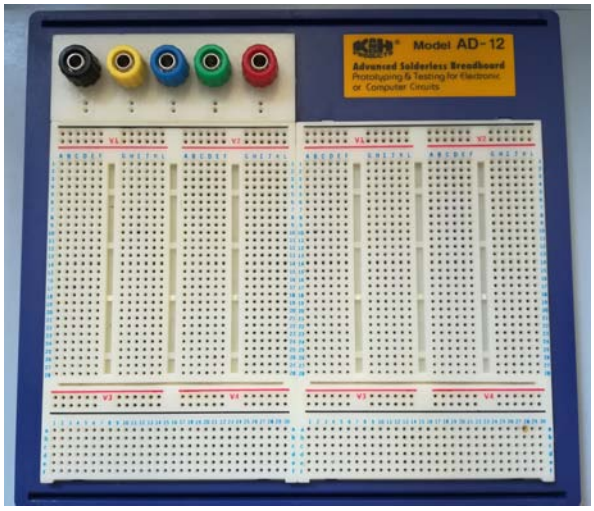
8.3.2 Bus strips

Used to provide power to the electronic components. A bus strip usually contains two rows: one for ground and one for a supply voltage. However, some breadboards only provide a single-row power distributions bus strip on each long side. Typically the row intended for a supply voltage is marked in red, while the row for ground is marked in black. Often the groups in a bus strip are indicated by gaps in the color marking. Bus strips typically run down one or both sides of a terminal strip or between terminal strips. On large breadboards additional bus strips can often be found on the top and bottom of terminal strips. Note there are two different common alignments for the power bus strips. On small boards, with about 30 rows, the holes for the power bus are often aligned between the signal holes. On larger boards, about 63 rows, the power bus strip holes are often in alignment with the signal holes. This makes some accessories designed for one board type incompatible with the other. There are no official standards, so the users need to pay attention to the compatibility between a specific model of breadboard and a specific accessory. Vendors of accessories and breadboards are not always clear in their specifications of which alignment they use. Seeing a close up photograph of the pin/hole arrangement can help determine compatibility. Some manufacturers provide separate bus and terminal strips. Others just provide breadboard blocks which contain both in one block. Often breadboard strips or blocks of one brand can be clipped together to make a larger breadboard. In a more robust variant, one or more breadboard strips are mounted on a sheet of metal. Typically, that backing sheet also holds a number of binding posts. These posts provide a clean way to connect an external power supply. This type of breadboard may be slightly easier to handle.

(a) dsPIC30F4011



(b) Breadboard



(c) Pickit 2

Figure 8.1: Main components used

8.4 Pickit 2

The PICkit 2 Development Programmer/Debugger (PG164120) is a low-cost development tool with an easy to use interface for programming and debugging Microchip's flash families of microcontrollers (figure 8.1c).

8.5 Procedure

In order to develope programs for the dsPIC30F4011 (briefly dsPIC) we use a plain text editor to write the C programs, Microchip's XC16 compiler to build them, and the PICkit 2 software application to download the compiled hex files onto the dsPIC. The process described below is Windows-specific:

1. create a new folder for every C written program. Try to save the code everytime with the word "main.c". The name of the folder may vary everytime a new program is typed;
2. compile the program with a simple build script which saves time avoiding to type build stripes every time that such operation is needed;
3. to run the build script, open a console window, navigate to the folder where your program files are located, and type the command "build.bat";

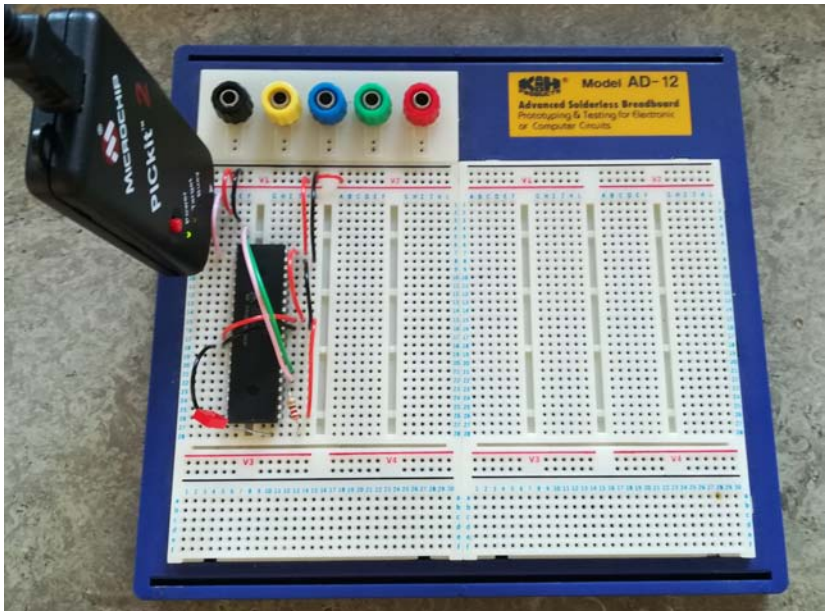


Figure 8.2: Simple application of a blinking LED implementation

4. assuming that the C program compiles without errors, two new files will be added to the directory, "a.out" and "a.hex";
5. make sure that the PICkit 2 is connected to the dsPIC circuit, and then launch the PICkit 2 application;
6. the file that has to be transfered in the flash memory on the dsPIC is"a.hex" and to download it onto the dsPIC, click the "Auto Import Hex + Write Device";
7. to run the program on the dsPIC, you may need to tick the V_{DD} check box to supply power to the circuit.

The figure 8.3 gives an idea of the overall process.

8.6 Simple example

For explaining what structure and pattern the system follows, we program to blink an LED on pin RD0 in the circuit. Pin RD0 is a specific pin that allows the user to have a digital output if it is properly set so. The figure 8.2 shows the real circuit obtained: it is assumed that the red lead brings the positive voltage (in this case $5V$) whereas the black (or gray) lead keeps a voltage equal to the ground reference ($0V$). On the top left hand we can see the employment of the PICkit2 that is connected straightforward to the USB socket working as a data relay as well as a power supply due to the low positive voltages used ($\simeq 5V$). Figure 8.4 shows more in detail which pins are connected and where exactly are. Almost every pin of the dsPIC can be used for different purposes being either an input or an output, designed either for analog or for digital signals and has the chance to be configured in a variety of different ways as shown in the datasheet in [7]. The figure 8.4 does not show all the possible outputs that a specific port can be because we want to give to the reader the simplest representation of the example in order to let him fully understands what has been done.

8.6.1 Breadboard circuitry

Examining figure 8.2 and 8.4 the pins used are more in detail:

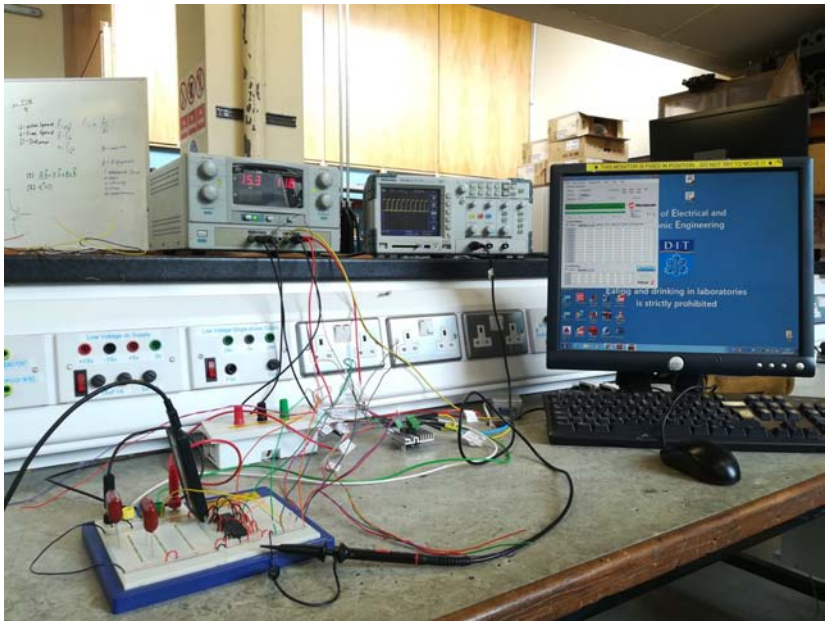


Figure 8.3: How to program a dsPIC

- *MCLR*: is the Master Clear or Reset. Most PICs have them, although not all PICs have the ability to handle it internally. In our PIC there is only the pin $n°1$ that works in such way. Basically this pin can be selected as an I/O pin versus the reset function. It is part of the chip's configuration;
- V_{DD}: is the positive supply for digital I/O pins. As depicted in figure 8.2 all the V_{DD} pins has to be connected with each other for having the overall system working;
- V_{SS}: indicates the ground for digital I/O pins. As for V_{DD}, even V_{SS} pins has to be connected between each other;
- *RD0*: represents a part of the PORTD pins that are bidirectional I/O pins;
- *PGD*: in-circuit serial programming data input/output pin which is essential for the transmission of data;
- *PGC*: in-circuit serial programming clock input pin that transmits the clock frequency previously imposed in the C script;
- AV_{SS}: positive supply for analog module and since there is no analog output coming out from the PIC this pin is linked with the other V_{SS};
- AV_{DD}: ground reference point for analog module that is connected to V_{DD} pins because of the absence of analog signals.

8.6.2 C Language programming

The C script illustrated in figure 8.5 is not so long thanks to the easiness of our goal (simple blinking). However, all the C language files comply with the following structure:

1. *header files*: a header file is a file containing C declarations and macro definitions to be shared between several source files. The inclusion of them is done with the C preprocessing directive "#include". Header files can be distincted in 2 types:

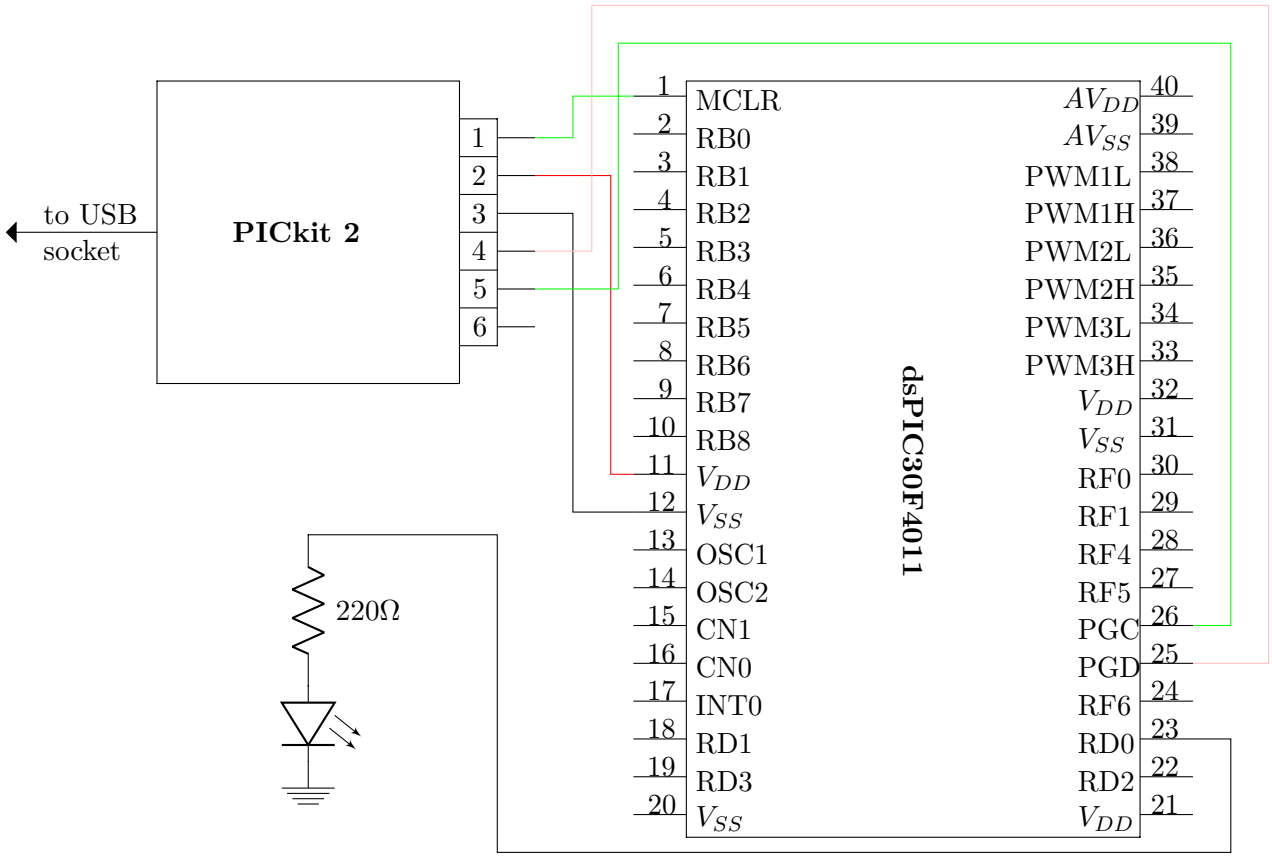


Figure 8.4: Schematic view of the realized circuit

```
//
// dsPIC30F4011 example - blink an LED on RD0
// Written by Mirco Seccardi, Last updated 22-6-2017
//

#include <xc.h>                  // header files
#include <libpic30.h>

// Configuration settings
_FOSC(CSW_FSCM_OFF & FRC_PLL16); // Fosc=16x7.5MHz, i.e. 30 MIPS
_FWDT(WDT_OFF);                  // Watchdog timer off
_FBORPOR(MCLR_DIS);              // Disable reset pin

void main()
{
    // Make RD0 a digital output
    _TRISD0 = 0;

    // Blink LED on RD0
    while(1)
    {
        _LATD0 = 1;
        __delay32(1000000);
        _LATD0 = 0;
        __delay32(5000000);
    }
}
```

Figure 8.5: Script written in C language downloaded into the the PICkit2

- *system header files* declare the interfaces to parts of the operating system. We include them in a program to supply the definitions and declarations needed to invoke system calls and libraries. These header files contain declarations for interfaces between the source files. Each time there is a group of related declarations and macro definitions all or most of which are needed in several different source files, it is a good idea to create a header file for them;
- *inclusion header files* produce the same results as copying the header file into each source file that needs them. Such copying would be time-consuming and error-prone. With a header file, the related declarations appear in only one place. If they need to be changed, they can be changed in one place, and programs that include the header file will automatically use the new version when next recompiled. With this trick we eliminate the labor of finding and changing all the copies as well as the risk that a failure to find one copy will result in inconsistencies within a program;

2. *configuration settings*: we see in figure 8.5 that 3 lines contain particular settings of the dsPIC; they respectively are:

   - *_FOSC(CSW_FSCM_OFF & FRC_PLL16)* which means that the frequency of the oscillator (FOSC) equal to $7.5MHz$ is multiplied by 16 which stands for the phase locked loop (PLL) that occurs 16 times in a cycle. Because our dsPIC30F4011 elaborates an instruction every 4 clock cycles, the amount of instructions per second are equal to
   $$f_{inst} = \frac{16 \cdot 7.5}{4} = 30MIPS \tag{8.6.1}$$
   where $MIPS$ means Millions of Instructions Per Second;
   - *_FWDT(WDT_OFF)* settles the *watchdog timer* off. A watchdog timer is an electronic timer that is used to detect and recover from computer malfunctions. During normal operation, the computer regularly resets the watchdog timer to prevent it from elapsing, or "timing out". If, due to a hardware fault or program error, the computer fails to reset the watchdog, the timer will elapse and generate a timeout signal. The timeout signal is used to initiate corrective actions that typically include placing the computer system in a safe state and restoring normal system operation;
   - *_FBORPOR(MCLR_DIS)* which turns the master clear reset (MCLR) off unabling the BOR and POR microcontrolling reset modes. In detail, the BOR consists of holding the microcontroller in reset state when the V_{DD} drops below a brown out threshold voltage. Not all the devices have BOR detection, but most do, and some have multiple voltage thresholds to select from. POR stands for Power On Reset and it is embedded upon power-up device experiences. POR voltage is always less than BOR voltage and between them the whole range of startup voltages can be covered to protect the device for proper operation after a power drop at the V_{DD} line.

3. *functions declaration*: all the function used in the script are now declared telling what sort of output we want (there is the chance to set a *void* output which means that there is nothing given by the function) as well as specify the name of it, the name of the function and what inputs (enclosed in brackets) are needed;

4. *main*: executes the heart of the C code written in the file by calling the functions previously prototyped;

5. *functions*: are a mixture of the ones called in the main and some others that do not need to be. They can have several purposes such as, imposing setup conditions, loading start data, reading an analog input, command an interrupt service routine (this runs without a main call). Generally, a setup function is always created imposing which pins are which in our dsPIC. Afterwards, almost everything depends on what we want to obtain. Particular attention is put on the setup function where all the commands are carried out afterwards that a careful study of the reference manual and data sheet has taken place. For instance, examining figure 8.5, the only setup made is about setting RD0 (pin 23) a digital output. The command TRISD0 in [8, 11-3] litterally reports:

   > *The TRISx register control bits determine whether each pin associated with the I/O port is an input or an output. If the TRIS bit for an I/O pin is a 1, then the pin is an input. If the TRIS bit for an I/O pin is a 0, then the pin is configured for an output. An easy way to remember is that a 1 looks like an I (input) and a 0 looks like an O (output). All port pins are defined as inputs after a Reset.*

   afterwards, with this condition of RD0 as a digital output (*whille(1)*), the while cycle in the main sets the range of time in which the pin has to be ON and OFF. Respectively

   - _LATD0=1

$$\_delay32(1000000) \rightarrow T_{on} = \frac{10^6}{30 \cdot 10^6} = 0.033s = 33ms \tag{8.6.2}$$

   - _LATD0=0

$$\_delay32(5000000) \rightarrow T_{off} = \frac{5 \cdot 10^6}{30 \cdot 10^6} = 0.166s = 166ms \tag{8.6.3}$$

The number 32 in the function name refers to the fact that the specified number of instruction cycles is a 32 bit unsigned value, allowing very long delays to be generated with a single call. Since the condition TRISD0=0 remains true for ever, the cycle automatically restarts over and over again.

8.7 Square wave form example

A more challenging example consists in the creation of 2 square waves with a shift angle between each other. This shift angle can be modified by setting specific values in the script and recompile everything next. The header files as well as the configuration settings are the same from the previous example. Moving into the main function, we start setting all 4 D ports (pins 23, 18, 22, 19) as digital output. It is curious to see the characteristic binary code used for defining TRISD. In fact, the first 2 digits establish that the following code is a binary number and not a decimal one as it is when nothing is declared. Consequently, TRISD sets the last 4 digits as digital outputs instead of the others recognized as digital inputs. Moving forward, there is the need to set the output compare which allow us to create the typical square shape at the output pins. For this reason, a dual compare match mode is exploited by OCM bits in the OCxCON registers; thus

$$OC1CONbits.OCM = 0b101 \qquad OC2CONbits.OCM = 0b101 \tag{8.7.1}$$

the quote in [8, 14-9] clearly says:

> *When control bits OCM<2:0> = 100 or 101 (OCxCON<2:0>), the selected output compare channel is configured for one of two Dual Compare Match modes which are:*

```
17  int main()
18  {
19      // Configure all four port D pins (RD0, RD1, RD2, RD3)
20      // as digital outputs
21      TRISD = 0b1111111111110000;
22
23      // Set OC channel 1 pulse start and stop times for one square-wave form
24      OC1R = 0;
25      OC1RS = 1500;
26
27      // Set OC channel 2 pulse start and stop times for the other square-wave form
28      OC2R = 1200;
29      OC2RS = 2700;
30
31    // Set output compare mode for continuous pulses
32      OC1CONbits.OCM = 0b101;
33      OC2CONbits.OCM = 0b101;
34
35      // Configure timer 2 (default timer for output compare)
36      PR2 = 3000; // 0.1ms period
37      T2CONbits.TON = 1; // Enable timer 2
38
39      // Flash an LED on RD0 indefinitely at 1Hz
40      while(1)
41      {
42          _LATD2 = 0;           // LED off
43          __delay32(30000000); // 500ms delay to view everything slower
44          _LATD2 = 1;           // LED on
45          __delay32(30000000); // 500ms delay to view everything slower
46      }
47
48      return 0;
49  }
```

Figure 8.6: Square wave C language for the dsPIC

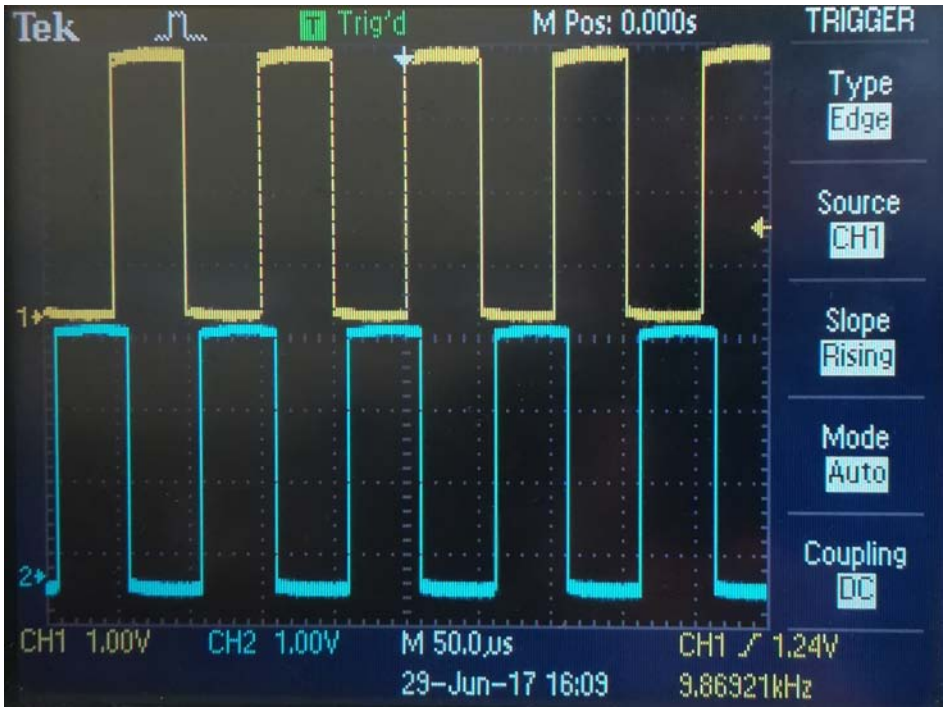


Figure 8.7: We can notice the displacement of $\frac{4}{5}\pi$

- *Single Output Pulse mode*
- *Continuous Output Pulse mode*

In the Dual Compare mode, the module uses both the OCxR and OCxRS registers for the compare match events. The OCxR register is compared against the incrementing timer count, TMRy, and the leading (rising) edge of the pulse is generated at the OCx pin, on a compare match event. The OCxRS register is then compared to the same incrementing timer count, TMRy, and the trailing (falling) edge of the pulse is generated at the OCx pin, on a compare match event.

Having set a timer equal to

$$PR2 = 3000 \rightarrow 100\mu s \tag{8.7.2}$$

with the start and finish pulses of the output compare equal to

$$OC1R = 0 \rightarrow OC1RS = 1500 \tag{8.7.3}$$

$$OC2R = 1200 \rightarrow OC2RS = 2700 \tag{8.7.4}$$

means a phase shift of

$$\varphi_{sw} = \frac{1200}{1500}\pi = \frac{4}{5}\pi \tag{8.7.5}$$

Finally, a delay of the LED blink has been set to 15000000 MIPS which corresponds to 0.5 seconds. This is made in order to let the reader know wheter the program is working correctly. In this case it does and a representation of the 2 square shapes is given in figure 8.7.

9 Sine wave form

We are now focusing on the creation of a sinusoidal voltage using a PWM. This is the classic use of the Pulse Width Modulation but nothing forbids it can be employed for different shapes. We can obtain a sine wave with 2 possible methods:

1. *discretization of the sine function* between 0 and 2π where the whole interval is split up in a certain amount of parts and every iteration that happens in the while cycle pushes the procedure forward step by step. When the final angle 2π is reached, settings make the while cycle starts all over again in an endless loop;

2. *usage of interrupt service routine* in which a specific counter register triggers an Interrupt Service Routine (ISR) inside of the dsPIC that compares the inner measurements and pop out with an instantaneous sine value. Afterwards the counter is reset and the implementation can go ahead to the next same counter register value. Tipically the values needed are gathered in an array outside the main script and the reference for calling it is possible by adding that specific array in the header files. The accuracy depends on the thickness of the steps between 2 consecutive array values: the smaller is the interval, the better the final shape will become.

9.1 Discretization of the sine function

Leaving the same header files and configuration settings explained in chapter 8, the main function starts with setting the pins for the PWM output. In fact, after a quick declaration for the configuration of the D ports as digital outputs, the code is shown in figure 9.1. In order of appearance we have:

- *PWMCON1* that enables the PWM pairs in complementary mode. Let is assume that an inverter is using these pins. Consequently, the switches in every leg must be commanded with two different signals namely one opposite to the other with the addition of the dead time as well. Examining the figure 9.1 when the PWMCON1 is being set, the instruction is not a typical number but it is written in hexadecimal code. This in-depth analysis is exposed in section 9.3;

- *PTCONbits.PTCKPS = 0* is the prescale value for allowing the f_{PWM} to be set even higher than what only a 15 bit number can do. There are 2 bits available that are thus available to give:

  - 11 = PWM time base input clock period is 64 T_{cy} (1 : 64 prescale);

```
void main()
{
    // Configure all 4 port D pins as digital outputs
    TRISD = 0;

    // Configure PWM for free running mode
    //
    //   PWM period = Tcy * prescale * PTPER = 0.33ns * 64 * PTPER
    //   PWM pulse width = (Tcy/2) * prescale * PDCx
    //
    PWMCON1 = 0x00FF;     // Enable all PWM pairs in complementary mode
    PTCONbits.PTCKPS = 0; // prescale=1:1 (0=1:1, 1=1:4, 2=1:16, 3=1:64)
    PTPER = 3000;         // 100us PWM period (15-bit period value)
    PDC1 = 3000;          //
    PDC2 = 3000;          //
    PDC3 = 3000;          //
    PTMR = 0;             // Clear 15-bit PWM timer counter
    PTCONbits.PTEN = 1;   // Enable PWM time base
    _DTAPS = 1;     // prescale value = 1:2 (0=1:1, 1=1:2, 2=1:4, 3=1:8)
    // DTA = dead time (3us) / (prescale* Tcy) = 45
    _DTA = 45;      // "_" use only for bits in a register 6 bits available so max=63

    float phase;
    int n=0, N=100;

    while(1)
    {
        n = n + 1;
        if (n == 100) n = 0;

        phase = n*2.0*3.14159/N; // angle in radians

        PDC1 = 3000 + 3000*sin(phase); // between 0 and 6000
        PDC2 = 3000 + 3000*sin(phase+2.0*3.14159/3.0); // between 0 and 6000
        PDC3 = 3000 + 3000*sin(phase+4.0*3.14159/3.0); // between 0 and 6000

        __delay32(1500);
    }
```

Figure 9.1: Main function of the discretization

  - 10 = PWM time base input clock period is 16 T_{cy} (1 : 16 prescale);
  - 01 = PWM time base input clock period is 4 T_{cy} (1 : 4 prescale);
  - 00 = PWM time base input clock period is 1 T_{cy} (1 : 1 prescale);

- *PTPER* stands for PWM Time Base Period Register that sets the counting period for PTMR (PWM Time Base Register). In other words it sets the length of the PWM period (f_{PWM} but as a dsPIC register value) which in our case is

$$PTPER = \frac{f_{cy}}{f_{PWM}(PTMRPrescaler)} = \frac{30 \cdot 10^6}{10^4 \cdot 1} = 3000 \tag{9.1.1}$$

- *PDCx* stays as a shorter representation for the duty cycles of each phase (PDC1, PDC2, PDC3). Theoretically it is between 0 and 1 but into the dsPIC it varies from 0 to $2 \cdot PTPER$. Having used a $PTPER$ equal to 3000, the correspondent duty cycle of 1 in the PDCx is equal to 6000 (see table 15-2 of the number 3 of the bibliography);

- *PTMR* enables the PWM time base (that increases one by one until PTPER) composed by a 15-bit timer with a prescaler and postscaler.

- *PTCONbits.PTEN* sets a specific number of bits that refers to PTEN register which turns the PWM time base on;

- *_DTAPS* is another way for changing a cluster of bits that respond to a specific register which is not shown in the code as it happens in the above-mentioned bullet. Here we set a prescale value that halves the one written in the next command line;

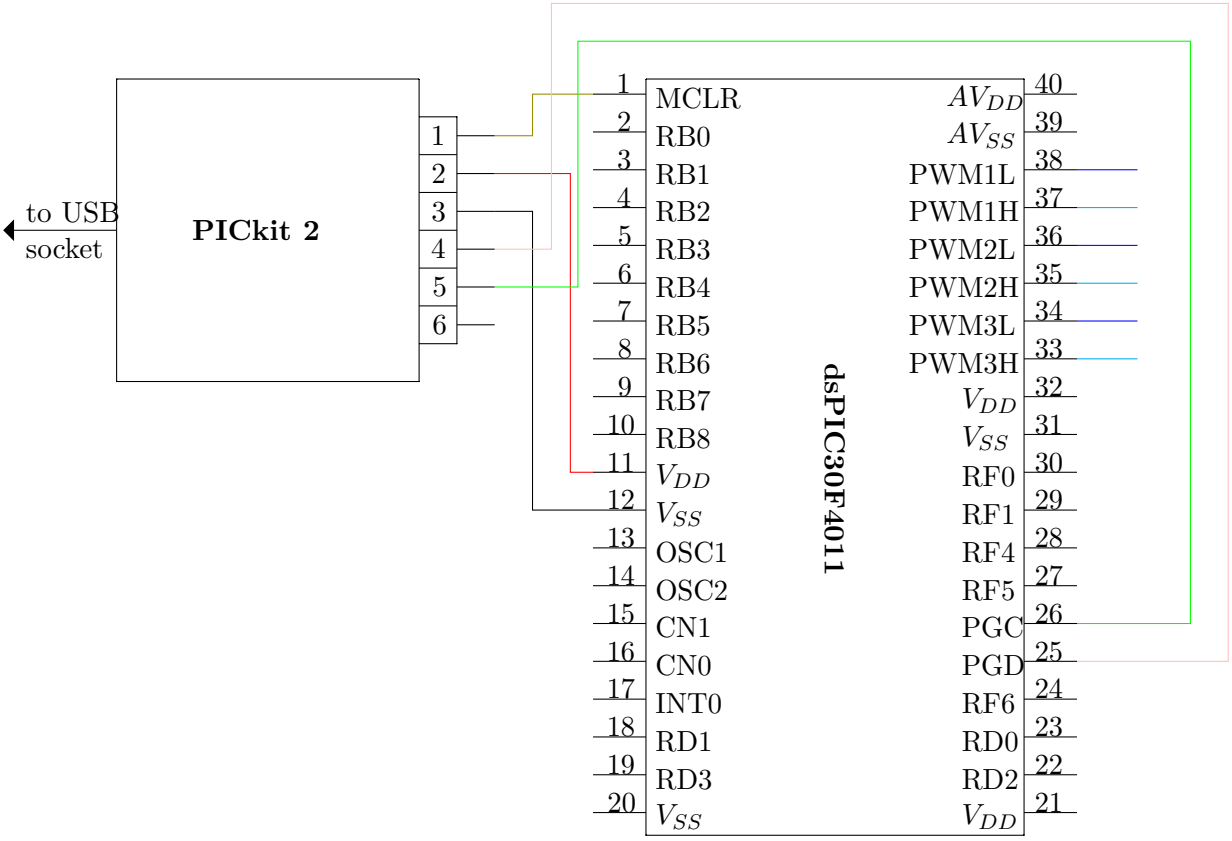


Figure 9.2: 3 Exploiting PWM dsPIC connections

- *_DTA* tells how long the dead time will be. With a prescaler value set on 1 : 2 (_DTAPS= 1), the number 45 must be thought as 90. The reason why a prescaler is here used is that a 6 bit can reach a cap of just

$$DTA_{max} = 2^n - 1 = 2^n - 1 = 64 - 1 = 63 \tag{9.1.2}$$

  thus, the equation 9.1.3 has to be taken into account

$$DTA = \frac{DT}{DTAPS \cdot T_{cy}} = \frac{3 \cdot 10^{-6}}{2 \cdot 33 \cdot 10^{-9}} = 45 \tag{9.1.3}$$

  where DT is the desired dead time wanted ($3\mu s$), DTAPS is the prescale value and T_{cy} is always equal to the instruction period;

- *float* imposes the floating point variables;

- *int* lists the integer variables.

Then the while cycle starts setting a true condition in brackets *"while(1)"*. Afterwards the *phase* is defined splitting it up in 100 intervals and the three duty cycles are calculated, one for each phase. At the end a delay is added for let the reader see what is going on. Physically the circuit connections are illustrated in figure 9.2. We can use a RC filter to verify that what the PWM has created is a truly sine wave. Therefore a RC impedance about

$$R = 220\Omega \tag{9.1.4}$$

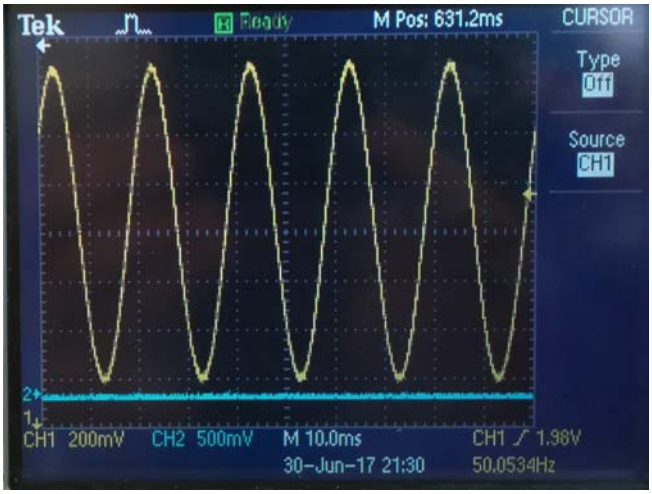


Figure 9.3: Sine wave between pin 38 and ground

$$C = 33\mu F \qquad X_{50Hz} = \frac{1}{2\pi 50C} = \frac{1}{2\pi 50 \cdot 33 \cdot 10^{-6}} = 96.46\Omega \tag{9.1.5}$$

$$X_{4850Hz} = \frac{1}{2\pi 50C} = \frac{1}{2\pi 4850 \cdot 33 \cdot 10^{-6}} = 0.99\Omega \tag{9.1.6}$$

is plug between one random PWM pin and the ground. The oscilloscope thus detects the figure 9.3 between the capacitor legs. This demonstrates the correctness of the C code uploaded into the non volatile dsPIC memory. A further procedure is to eliminate the PICkit2 connection and to use a battery pack as a source. This is widely adopted because of convenience since there is no need to write or modify the definitive program memorised inside the dsPIC unless while it is being built up. The battery is made by 4 AA $1.5V$ sources each connected in series. It has to be said that normally the nominal voltage that the dsPIC deals with is $5V$ but we discovered that it can withstand, dissipating a higher quantity of heat onto its surface, up to $6V$. Although this level does not remain as stable as the PICkit2 source did because of the greater current circulating in the overall circuit, a PWM as well as a filtered sine wave (figure 9.3) are still achieved. Also a discharge of the battery pack should be taken into account since it will lead to lower voltages than what we have detected on the first day of trials.

9.1.1 RC Filter

The values of resistance and capacitance in 9.1.4 and 9.1.5 used for filtering the PWM are not chosen randomly but conventionally are found with this pattern:

1. calculating the pulsation ω_{filter} having the fundamental frequency and the PWM frequency known.

$$\omega_{filter} = 2\pi\sqrt{f \cdot f_{PWM}} = 2\pi\sqrt{50 \cdot 10000} = 4442.88\frac{rad}{s} \tag{9.1.7}$$

2. assuming the condition $R = \frac{1}{\omega C}$

3. according to the capacitors at our disposal (in our case $33\mu F$) the resistor must be

$$R = \frac{1}{\omega_{filter} C} = \frac{1}{4442.88 \cdot 33 \cdot 10^{-6}} = 6.82\Omega \tag{9.1.8}$$

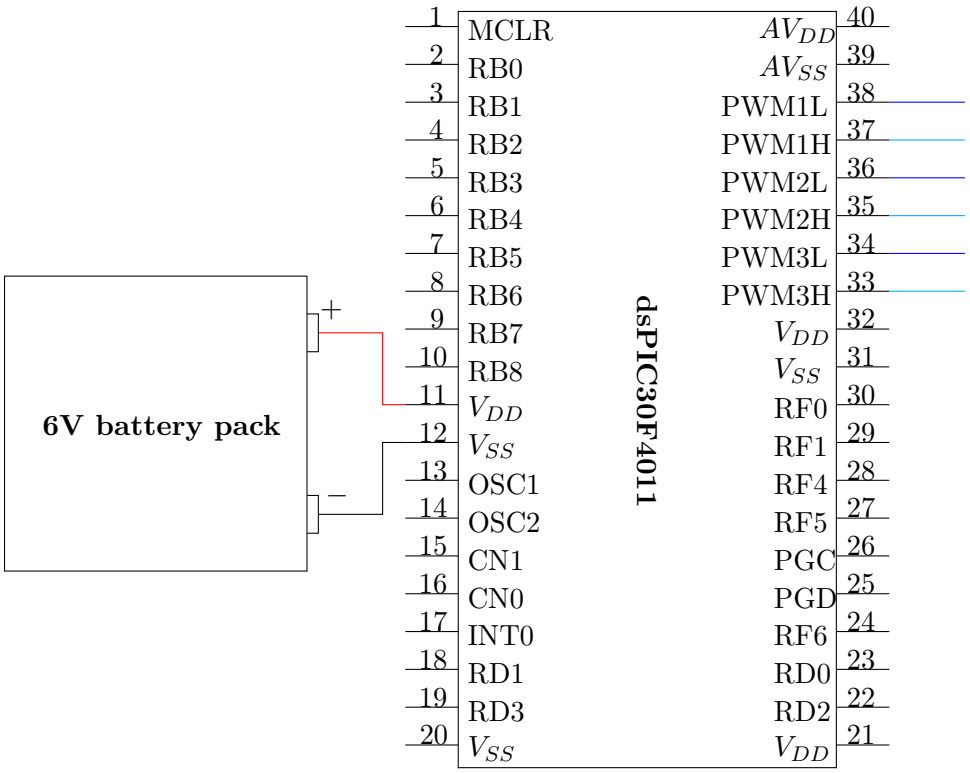


Figure 9.4: Implementation with the battery pack using the flash memory of the dsPIC

4. at this point we evaluate if the resistor is high enough to limit the high harmonic currents for having a clear sine wave through the capacitor. if it is not so or if it is too small that it does not block anything, the experience dictates the right amount of resistance needed.

As seen in figure 9.3 there is an offset on the channel of the detected sine pace. This is due to the rate of resistance used in the filter since the capacitor acts as an open circuit for DC voltages and currents. The higher the resistance is, the less the offset would be because of the increased drop voltage across the resistor. A representation can be found in figure 9.5b. A Ohmic-inductive filter could also be used where this time, the sinusoidal waveform will be acquired across the resistor. Figure 9.5a gives an idea. The two fallbacks are collected in figure 9.5.

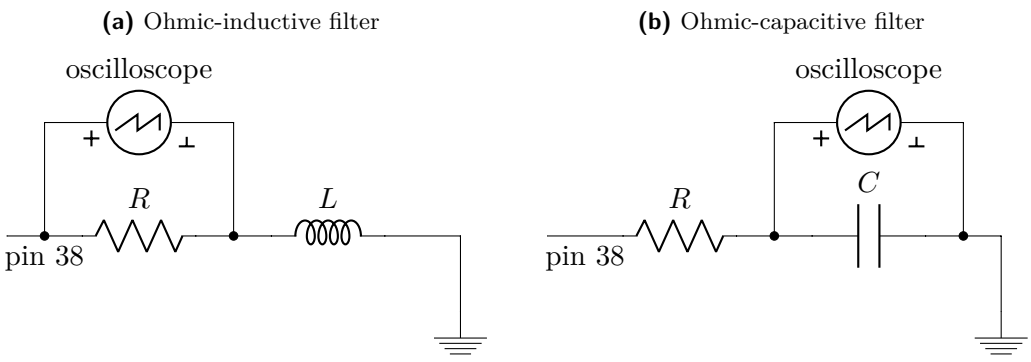


Figure 9.5: Possible filters used

9.2 PWM Interrupt employment

Another way for accomplishing our purposes is shown by the interrupt service routine dedicated to the creation of the PWM cycles. To achieve this, the *_PWMIE* bit (Pulse Width Modulation Interrupt Enable bit) present in the *IEC2* (Interrupt Enable Control Register 2) has to be set to 1. The circuitry in figure 9.2 is still valid. An interrupt service routine must be created and it represents the core of the PWM generation coming out from the dsPIC. In detail we have:

- *void__attribute__((interrupt, auto_psv))_PWMInterrupt(void)* is the declaration of the interrupt service routine. For further explanation of the code, the reader is invited to read [8];
- *__PWMIF* clears the interrupt bit flag. It determines whether or not the CPU will handle maskable hardware interrupts. With the declaration

$$PWMIF = 0 \tag{9.2.1}$$

  the interrupt has not occured. However, we are already inside the ISR (Interrupt Service Routine). As a result, the script must proceed. This flag becomes useful for the next interrupt occuring the next PWM cycle;
- *float* and *int* variables namely floating variables and integer ones. Float is a lighter declaration than *double*. As the name implies, a double has 2x the precision of float. In general a double has 15 decimal digits of precision, while float has 7. There are differences between the two even when we speak of big numbers such as the biggest number possible that is taken as infinity which is:
  - $3 \cdot 10^{38}$ for float;
  - $1.7 \cdot 10^{308}$ for double.

  This precision loss could lead to truncation errors much easier to float up. Since we do not deal with either huge numbers or small ones, those two types of numbers could be considered as the same;
- *static int* and *static double* which retains the close loop end value of the ISR and starts back again without being reset to zero. This is exploited for the variable time t and the duty cycle *d1* and *d2*;
- the first case is when the first interrupt is running. The internal values of the duty cycles $PDC1$ and $PDC2$ are imposed zero. This is because the static value of *counterint* is starting with zero and thus the initial conditions is zero as well;
- the most common case is about setting the internal $PDC1$ and $PDC2$ respectively equal to $d1$ and $d2$;
- an "*if*" cycle then starts creating an angle, adding a delta and transforming the "*phasea*" into a number between 0 and 4000 to refers to a look up sine array and eventually updating the duty cycle;
- reaching $t = 0.2$ determines a reset in the variable and the "*if*" cycle starts back again.

The interrupt service routine used is illustrated in figure 9.6.

```
void __attribute__((interrupt, auto_psv)) _PWMInterrupt(void)
{

    _PWMIF = 0;                 // Clear interrupt flag

    static int counterint=0;    // value from the previous cycle is needed
    float angle1;
    int xa, ratioa;
    double V, phasea;
    static int d1,d2;
    static double t=0.0;

    //first interrupt no measures given therefore duty = 0
    if (counterint == 0)
    {
        PDC1 = 0;
        PDC2 = 0;
    }
    else
    {
        PDC1 = d1;
        PDC2 = d2;
    }


        ////////////////////////////
        _LATD0 = 1; // debug pin on
        ////////////////////////////

        // PART 1 //

    if ( t<0.02 )
    {
        angle1 = 2.0*PI*50.0*t;


        // PART 2 //


        phasea = (angle1) + delta;


        // PART 3 //

        // according to the look up table choose the right sine value for the phase A
        ratioa = phasea/deltaarray;
        if (ratioa == 4000) ratioa = 0;
              xa = ratioa;                          // position on the look up table
              V = sarray[xa];                       // taking the value as new voltage
        d1 = 0.95 * (1.0 + V ) * PTPER ;            // first leg
        d2 = 0.95 * (1.0 - V ) * PTPER ;            // second leg

        // updating the counter
        t = t + 0.0001;
        counterint= counterint + 1;

        ////////////////////////////
        _LATD0 = 0; // debug pin off
        ////////////////////////////

    }
    else t=0;
}
```

Figure 9.6: Interrupt service routine for the creation of the PWM

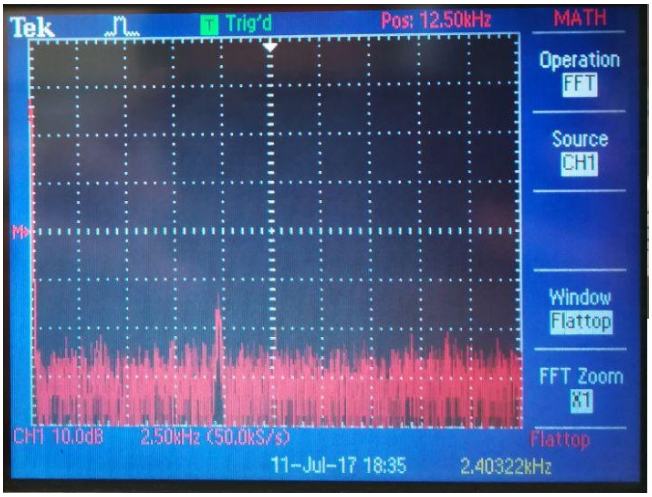


Figure 9.7: PWM FFT Spectrum of the output pin

9.2.1 Enhancement of the wave form

The important improvement done for the PWM outuput regards the change in the PWM period or f_{PWM}. We know that for a clear spectrum with the less noise possible, in a single phase full bridge, the modulation frequency m_f which is defined as the ratio between the PWM frequency and the fundamental

$$m_f = \frac{f_{PWM}}{f_{50Hz}} = \frac{10000}{50} = 200 \tag{9.2.2}$$

has not to be neither a multiple of 3 nor an even number, as happened in the previous section (sine wave discretization) with the value of the 9.2.2, but an odd non multiple of 3. As a result, it has been chosen

$$m_f = 197 \qquad \rightarrow \qquad f^*_{PWM} = f_{50Hz} \cdot m_f = 50 \cdot 197 = 9850Hz \tag{9.2.3}$$

As a consequence, the PTPER varies and its final value, according to the 9.1.1, corresponds to

$$PTPER^* = \frac{30 \cdot 10^6}{9850} = 3045 \tag{9.2.4}$$

achieving the spectrum in figure 9.7. In detail, the magnitudes of the relevant harmonics orders are

$$M_{50Hz_{dB}} \simeq 60dB \qquad M_{9850Hz_{dB}} \simeq 13dB \tag{9.2.5}$$

where dB stands for decibel equal to

$$M_{dB} = 20 \log M_{linear} \qquad \rightarrow \qquad M_{linear} = 10^{\frac{M_{dB}}{20}} \tag{9.2.6}$$

consequently

$$M_{50Hz_{linear}} = 10^{\frac{60}{20}} = 1000 \qquad M_{9850Hz_{linear}} = 10^{\frac{13}{20}} = 4.46 \tag{9.2.7}$$

hence the percentage of the distorsion is around 0.5%.

10-**base number**	10	11	12	13	14	15
16-**base digit**	A	B	C	D	E	F

Table 9.1: Equivalence for the last 6 digits of the hexadecimal system with the decimal numbers

9.3 Hexadecimal code

In the programming environment it is widely known the employment of the binary code, composed by only two possible figures i.e. 0 and 1. With these two figures we explicitly link the number 1 with an ON state. Meanwhile the number 0 with an OFF state. However, there are other codes that could be encountered and useful at the same time. These codes are:

- *octal* is the base-8 number system that contains the figures $0, 1, 2, 3, 4, 5, 6, 7$;
- *hexadecimal* where there are 16 digits available that are 0 to 9 and the table 9.1 adds the correspondence for the decimal numbers between 10 to 15.

As we seen the last digits are made by letters up to F which conveys the greatest value. This code becomes really convenient when a shorter representation of a binary code is required. For example, taking the hexadecimal number where the convention for expressing it as a base-16 number is to put in front of the number the expression "$0x$" as well as "$0b$" for the binary code, we obtain

$$\mathbf{0x} \quad B6A \rightarrow \mathbf{0b} \quad 1011 \quad 0110 \quad 1010 \tag{9.3.1}$$

$$\mathbf{0x} \quad FCB4AD \rightarrow \mathbf{0b} \quad 1111 \quad 1100 \quad 1011 \quad 0100 \quad 1100 \quad 1101 \tag{9.3.2}$$

It is drawn that 4 bits correspond to an only 1 digit in the hex system. Furthermore this is always true since 16 (digit number in the hexadecimal) is a power of 2 (0 and 1 implemented in the binary code) and no particular effort for the conversion between the 2 representations is required. For completing the treatise, the 9.3.1 in decimal base "*0d*"is equal to

$$B \rightarrow \mathbf{0d}11 \cdot 16^2 = 2816 \quad 6 \rightarrow \mathbf{0d}6 \cdot 16^1 = 96 \quad A \rightarrow \mathbf{0d}10 \cdot 16^0 = 10 \tag{9.3.3}$$

$$\mathbf{0d} \quad 2816 + 96 + 10 = 2922 \tag{9.3.4}$$

the transformation of the 9.3.2 is let to the reader as exercise.

9.4 Various implementations and issues encountered

From C language script in figure 9.6, many others PWM could be created for example depending either on the number of phases (single, three phase or even tetraphase) or on what type of inverter it is available (half bridge, full bridge). In this report, a modulation index $m_a = 0.95$ has been adopted but nothing impedes to set different values of m_a varying between 0 and 1. Obviously, this script can be run using the battery pack instead of the PICkit2 into the USB socket too (figure 9.4). In addition, other applications of the PWM can be designed with the Interrupt Service Routine (ISR) as the injection of unbalanced current in order to erase the grid unbalances. A shortcoming that frequently appears during programming is about the computation time that the dsPIC takes for carrying out all the C code that is compiled inside of the memory (figure 9.1 for instance). There are several hints to let the reader achieved the least computation time into the dsPIC or inside a CPU in general:

- reduce the "*if*", "*while*", "*for*" cycles to the minimum amount possible since they imply that the CPU should verifies a certain condition first and this is time consuming;
- use external files containing useful arrays (here a sine array is being used) in order to let the main program free from any assesment constraint;
- avoid printing lines and superficial commands that should organize better the overall work.

At the end, if we do not have a computation time less than the PWM frequency, the effect is seen into a fundamental frequency equal to half, one third or less of the desired one. Therefore the only solution might be provided by lessening the f_{PWM} waiving a few accuracy skills that bring to a shape decrease even though slightly affecting the harmonic spectrum.

10 Inverter implementation

In this chapter we deepen our studies in the communication and transmission of the gate pulses to the inverter for finally obtaining the real conversion from the DC voltage to a PWM shape wave form which we feed the load with. The inverter is the last component of the final scheme and it will be here analized focusing on its characteristics and the best way to use them.

10.1 Type of inverter

We use the *STGIPS10K60T* model due to the lower cost and simplicity in the layout. Its cost is near 16 € and this brings a huge availability inside of the laboratory because it is affordable and can be bought in huge quantity all in once. If something wrong happens causing the failure of the piece, it will be replaced without taking care of the economical impact. This helps even though the reader has never put the hands on any kind of inverter. Managing cheap power electronics is the best way to start for learning how that specific type of component behaves even though it has been previously studied. It is divided in two set of pins, each for every long rail. The rail with most of the pins is the *control side* because is entirely dedicated to the digital input that will be elaborated for the gate commands. Whereas the other rail is the *power side* quickly recognized by the thicker pins used for delivering power to the load (figure 10.1, 10.3a, 10.3b). The general features of the inverter are:

- *IPM (Intelligent Power Module) 10A, 600V, 3 phase IGBT (Insulated Gate Bipolar Transistor) inverter bridge including control ICs (Integrated Circuits) for gate driving and freewheeling diodes* generally all the inverters are made for 3 phase purposes unless otherwise specified but there is no problem in using 1 or 2 phases out of 3. Furthermore the integrated circuits depicted in figure 10.2 take care not only of gate driving but even of dead time generation;
- *short-circuit rugged IGBTs* where the switches can withstand strong short circuits currents;
- V_{CE}*(sat) negative temperature coefficient* explains that the collector emitter voltage in saturation conditions has a negative slope. This enhance the conductivity through the switches because it decreases the ON state internal resistance;
- *3.3V, 5V, 15V CMOS/TTL inputs comparators with hysteresis and pull down/pull up resistors* noting different types of signals for the control side pins in comparison with what used in this document;

- *undervoltage lockout* is an electronic circuit used to turn off the power of an electronic device in the event of the voltage dropping below the operational value;
- *internal bootstrap diode* used for charging and discharging process of the bootstrap voltage (see section 10.3);
- *interlocking function* it makes the state of two mechanisms or functions mutually dependent. It is used in the creation of the dead time injecting the same shape for the high and low side but turn the signal the way around automatically as for the low switches;
- *shut down function* is dedicated to one pin that when an unpredictable problem occurs, it shuts everything down automatically avoiding further damage to the internal circuits;
- *DBC (Direct Bonded Copper) substrate leading to low thermal resistance* this technology has a very good thermal conductivity. It is composed by a ceramic tile with a sheet of copper bonded to one or both sides. Upper copper usually is performed in order to being part of an electrical circuit;
- *isolation rating of* $2500\frac{V}{min}$;
- $5k\Omega$ *NTC (Negative Temperature Coefficient) for temperature control* can effectively limit inrush current. It provides variable resistance based on its temperature. At first, the initial temperature of the alleged NTC thermistor is low, providing high resistance. When the system is powered on, it energizes the NTC thermistor, causing the temperature to rise, and thus lowering the resistance. As resistance drops to a low value, the current passes through without adversely affecting normal operation or power efficiency. For example, let is consider a system with $10A$ continuous current and an inrush current of 100 A while the NTC only allows $35A$ to pass through. Upon power up, an NTC thermistor has an initial resistance of 10Ω. Then, as the NTC self-heats, its resistance drops and lowers the current until the inrush current is over. The NTC still continues to heat, dropping resistance to as low as 0.05Ω where it reaches a steady state and passes current through minimum loss in efficiency.

The first modification to do is due to the uneven pattern of both the control pins and power pins. Because the control pins do not fit into the breadboard, we use connectors that create an electrical continuity between each pin and its relative lead that can thus be plug into the breadboard holes (figure 10.1). However, the pin 16 is left away since a single connector package can gather only 5 pins and even because it represents the NTC which is redundant for our purposes in this report. As for the power side, the same pattern is carried out with thicker cables, leaving the pins 19 and 22 unpluged because already internally connected with the pin 25 (figure 10.2). Studying the scheme in figure 10.2, we notice an RC circuit connected between DT and GND which is used to implement dead time. All the pins physically accessible are put outside the greater black square and the table 10.2 explains which pin is which. The figure 10.3a shows where all the pins numbers are respectively reffered to. About the size, figure 10.3b shows that everything is contained between $6cm$ in width and $5cm$ in height because of the metal slab embedded to the inverter that helps to dissipate the heat that becomes significant from a current flowing of $1A$. Roughly speaking, the inverter itself is around the dimension of $4cm$ wide and $3cm$ high.

10.2 Installation

For a correct operation, two DC voltage sources are required using one for the control side (pin 1 to 16) and the other for the DC link voltage during the *test conditions* as the DC

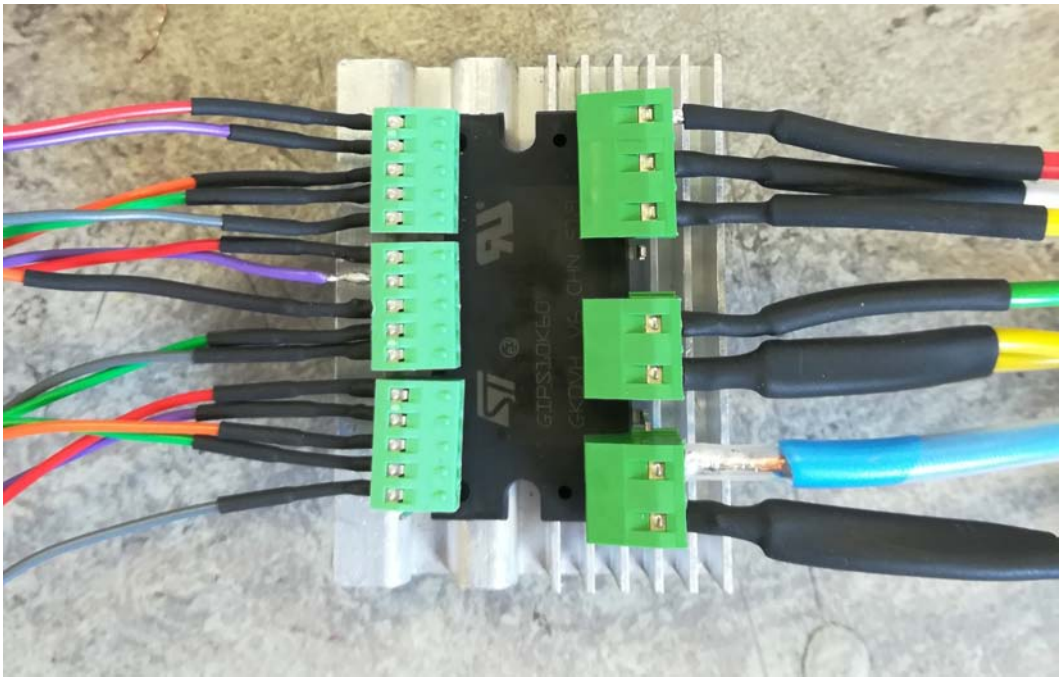


Figure 10.1: Inverter pins popped into the connectors

Symbol	Parameters	Conditions	Value			Unit
			min	typ	max	
V_{PN}	supply voltage	$P - N_U, N_V, N_W$	–	300	400	V
V_{CC}	control supply voltage	$V_{CC} - GND$	13.5	15	18	V
V_{BS}	high side bias voltage	$V_{boot_i} - OUT_i$ for $i = U, V, W$	13	–	18	V
t_{dead}	dead time	for each input signal	1	–	–	μs
f_{PWM}	PWM input signal	$-40°C < T_{junction} < 125°C$	–	–	20	kHz

Table 10.1: Recommended operating conditions

power supply can be boosted up to $30V$. It is also recommended an electrolytic capacitor attached to the DC link when the inverter will be tested at the rated values since it smoothens the DC input ripples. As for the low voltage power supply (pin 5), it boosts the incoming low and high input of $5V$ up to tipycally $15V$ charging the boostrap capacitor as well. The bootstrap voltage is discussed in detail in the following section. Moving to the other pages of the GIPS10K60T data sheet there are plenty of different tables gathering maximum ratings for both the control and power side as well as different tests carried out that shows the basics of what such a component does. For testing, a low voltage is advised (not over $50V$) while we can neglect most of the recommendations. When everything has been checked and a good behaviour is seen, the recommended measurements has to be followed (table 10.1) and they should be maintained during the steady state regime. Other recommendations which are preferable to comply with are:

- input signal HIN is active high logic. A $85k\Omega$ (typ.) pull down resistor is built-in for each high side input. If an external RC filter is used, for noise immunity, pay attention to the variation of the input signal level;
- input signal $\overline{LIN}$ is active low logic. A $720k\Omega$ (typ.) pull-up resistor, connected to an internal $5V$ regulator through a diode, is built-in for each low side input;

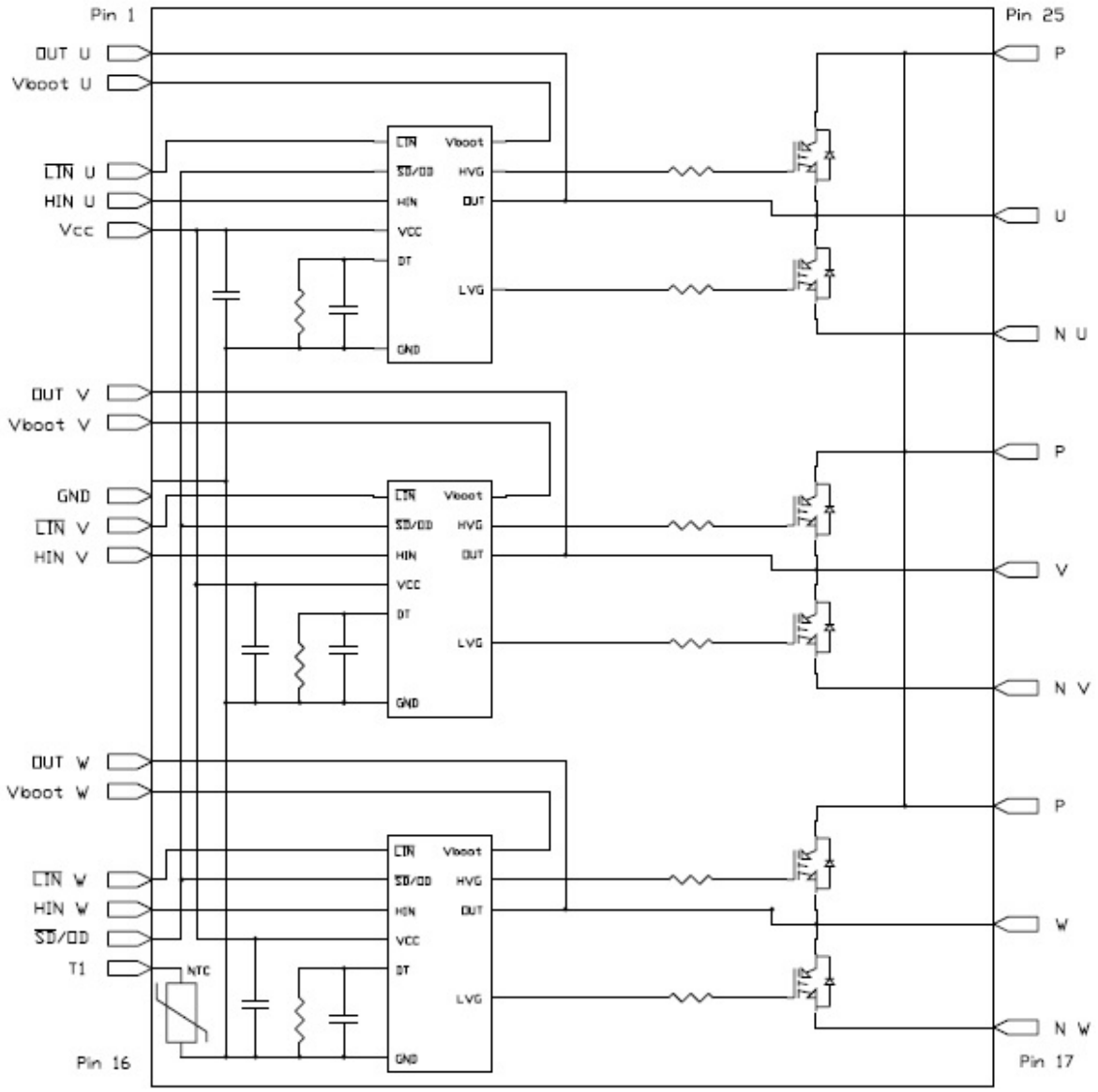


Figure 10.2: Internal block diagram

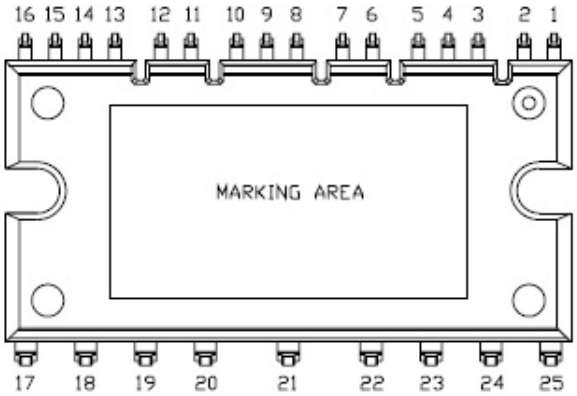
- to prevent the input signals oscillation, the wiring of each input should be as short as possible;

- each capacitor should be located as nearby the pins of IPM as possible;

- the $\overline{SD}/OD$ signal should be pulled up to $5V/3.3V$ with an external resistor.

Nevertheless this procedure does not assure that the component will not be broken by any human unpredictable misunderstanding.

(a) Pin layout



(b) Inverter measurement

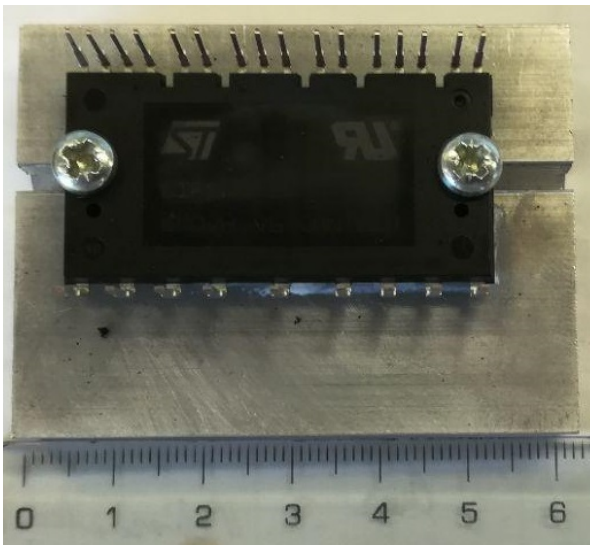


Figure 10.3: Inverter used in the project

Pin n°	Symbol	Description
1	OUT_U	High side reference output for U phase
2	V_{boot_U}	Bootstrap voltage for U phase
3	$\overline{LIN}_U$	Low side logic input for U phase
4	HIN_U	High side logic input for U phase
5	V_{CC}	Low voltage power supply
6	OUT_V	High side reference output for V phase
7	V_{boot_V}	Bootstrap voltage for V phase
8	GND	Ground
9	$\overline{LIN}_V$	Low side logic input for V phase
10	HIN_V	High side logic input for V phase
11	OUT_W	High side reference output for W phase
12	V_{boot_W}	Bootstrap voltage for W phase
13	$\overline{LIN}_W$	Low side logic input for W phase
14	HIN_W	High side logic input for W phase
15	$\overline{SD}/OD$	Shut down logic input (active low)/open drain (comparator output)
16	T_1	NTC thermistor terminal
17	N_W	Negative DC input for W phase
18	W	W phase output
19	P	Positive DC input
20	N_V	Negative DC input for V phase
21	V	V phase output
22	W	Positive DC input
23	N_U	Negative DC input for U phase
24	W	U phase output
25	P	Positive DC input

Table 10.2: Pin description

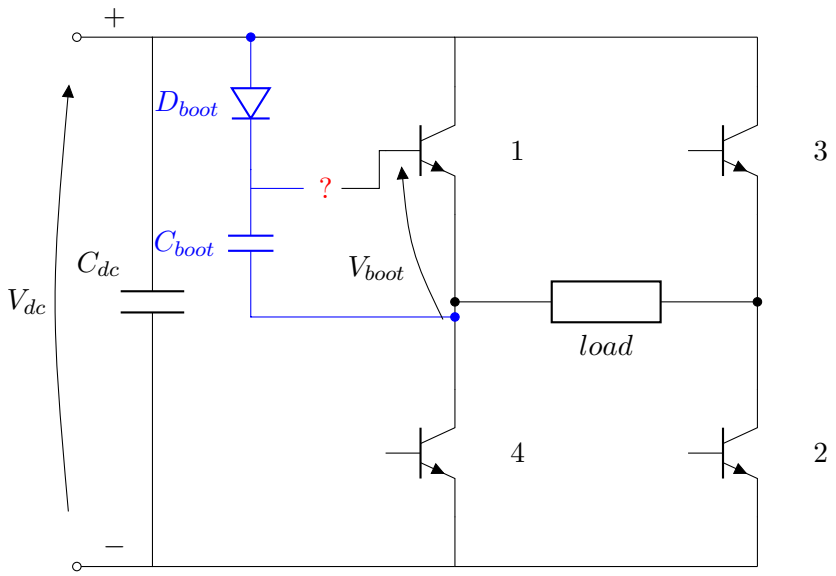


Figure 10.4: Bootstrap capacitor effect and triggering of the high side switches

10.3 Bootstrap voltage

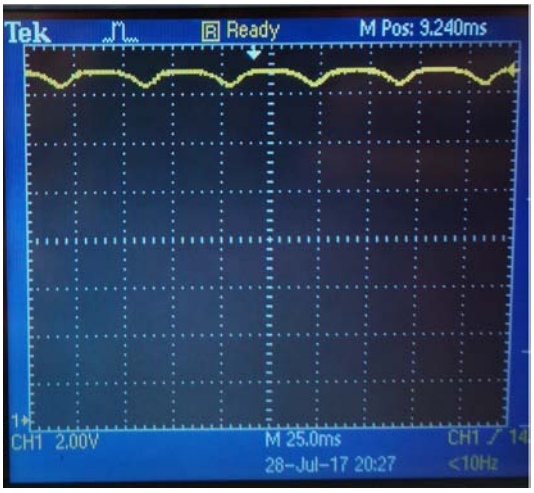
The bootstrap voltage is tipycal of every inverter since it works as the trigger for the high side switches because during the commutation from HIN_x OFF $\rightarrow$ HIN_x ON it gives a higher voltage in order to activate the conduction that it would not be happening if the same level of the + terminal of the DC link is used. Truly speaking the inner circuitry of the inverter dampens the positive voltage of the bootstrap capacitor leaving it always $\simeq 15V$. In figure 10.4 only one bootstrap capacitor is depicted but there is a mutual circuitry correlated with the activation of the IGBT number 3 depending on the state of the switch 2. In order to explain the operation of the bootstrap effect, only the first leg with the IGBT number 1 and 4 of figure 10.4 is taken into account. The steps are as follows:

1. starting with $n°1$ off means that $n°4$ is conducing with a drop voltage equal to $0V$;
2. meanwhile the diode D_{boot} is conducting and the C_{boot} is being charged up to a voltage of $12V$;
3. once the commutation takes place in $n°1$ the ON state of $12V$ is added with the voltage of the bootstrap equal to $-12V$ leading to a total voltage for the gate equal to $V_{boot} \simeq 24V$;
4. the diode is reverse bias and thus the gate is activated;
5. by the time $n°1$ has finished to conduct the process can restart from the beginning.

The reason of the question mark in figure 10.4 is due to the fact that the connections between the boostrap capacitor and the gate can be found nowhere but inside the company that makes the inverter. It is impossible with our means to proceed further and only a general explanation of the phenomenon can be given. Other attention can be drawn to the pace detected for the bootstrap voltage through the oscilloscope:

- the figure 10.5a illustrates how the sinusoidal output affects the constant level of $V_{boot} = 15V$. These peaks and troughs are down to the natural shape of the fundamental. The maintanance of a constant V_{boot} is challenging since the charging intervals during the minimum values are really short because of the low state switch conductions. On

(a) Overview of the bootstrap voltage



(b) PWM effect on the bootstrap voltage

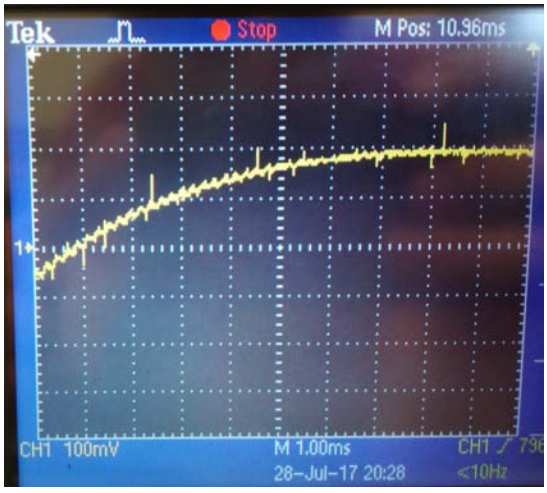


Figure 10.5: Bootstrap voltage

the other hand, during the maximum peak the bootstrap capacitor has higher times of charging and its voltage rises again.

- a detailed sigth is offered by figure 10.5b where the microscopic changes of the voltage are due to the PWM cycles: the duty cycle gives a rise in V_{boot} when the $n°1$ is on, whereas a fall when $n°1$ is off. This changes can be seen throughout the fundamental period reaching a duty cycle around 1 corresponding with the maximum peak, and $\simeq 0$ when the negative fundamental peak value is near by.

11 Overall trial

What has been described in the previous chapters is now gathered altogether and verified in order to obtain a complete functioning and self maintained system. The model that we wish to phisically realise (showed in figure 11.1 and already analised in section 7.8) is unfortunately too much complex in terms of knowledges possessed by the writer as well as in terms of time required to succeed in comparison with the mere simulation done on MATLAB/SIMULINK. What this chapter will describe regards a step by step process so as to arrive to the real complete test that has been carried out in the laboratory on an equivalent circuit.

11.1 Close loop issues

For obtaining a close loop control (figure 11.1), the C language employed has to be much more complicated requiring not only a deep knowledge of the C programming but even a new dsPIC in the event that all the computations would be high time consuming. In addition, a detection of the grid voltage (phase and amplitude) must be fed into the close loop. Consequently, the close loop realisation is definitely put aside and an open loop control seems feasible (figure 11.2).

11.2 Single phase active grid connection

Furthermore, in order not to create unbalances in the single phase active grid, the means are not taken into account, delimitating our employment in feeding a single phase passive load (figure 11.3) where self excitation capacitors are vital in this step because there is no active load anymore. This is down to the fact that, in order to deliver our active power without injecting reactive power, a certain shift angle has to be guaranteed. Therefore, we should know the grid angle and then try to put in parallel our voltage instantaneously. We do not do this primarily for safety reason and secondly for time restrictions. It has to be said that the real goal of our trial would be to see if the inverter can withstand voltages and currents below the rated values.

11.3 Generator absence

Because of the redundancy of the open loop control, the generator, the self excitation capacitors and the rectifier are substituted with a single variable DC voltage power supply (figure 11.4). This gets rid of all the problems that a self excitation must give so as to either absence of

$V_{dc_{variable}}[V]$	$V_{load}[V]$	$m_{a_{real}}$
20	13	0.87
40	25	0.87
60	36	0.87
80	50	0.88
100	61	0.88
120	75	0.88
140	90	0.9
160	100	0.9
180	125	0.98
200	139	0.98
220	155	0.98
240	*burst*	*burst*

(a) First trial

$V_{dc_{variable}}[V]$	$V_{load}[V]$	$m_{a_{real}}$
20	13	0.87
40	25	0.87
60	36	0.87
80	50	0.88
100	61	0.88
120	75	0.88
140	90	0.9
160	100	0.9
180	115	0.91
200	128	0.91
220	142	0.91
240	156	0.92
260	170	0.92
280	187	0.92

(b) Second trial

Table 11.1: Trials of the project depicted in figure 11.4. $280V$ is the maximum $V_{dc_{variable}}$ possible

residual magnetism (need to magnetise the windings by flash poles or somewhat) or imperfect self-excitation capacitance value.

11.4 Laboratory trials

Figure 11.4 gathers altogether the various components described in the previous chapters and illustrates the connections for the first verification of the inverter capability. As we see in figure 11.4, the DC voltage variable supply feeds the DC link of the inverter and a PWM single phase output is obtained. For the *open loop control*, $15V$ on the inverter command rail are required as well as $5V$ supply for the dsPIC creating a loop of PWM pulses leading to a sine wave shape as fundamental. Everything is connected through proper leads that sustain a value of maximum current higher than what circulates in the model. A breadboard is still employed between the dsPIC and the inverter. During all the procedure the current did not go over the rate of $1A$ and the resistance remains around 220Ω.

11.4.1 First trial

Only the first and second leg of the inverter are deployed leaving the third one floating on both the command and power side of the STGIPS10K60T. The verification consists of rising the $V_{dc_{variable}}$ by steps of $20V$ detecting in the meantime what values on the rest of the circuit are being obtained. Baring in mind a modulation index m_a set at 0.95 (as figure 9.6 quotes), the real modulation index $m_{a_{real}}$ is also calculated through annotation of V_{load} (RMS value) and $V_{dc_{variable}}$. The table 11.1a stops at $V_{dc_{variable}} = 220V$ since afterwards a burst of an inner IGBT happened. The reason is allegedly down to the third phase floating voltage that can have produced some additional voltage and thus did not comply with the voltage inverter features exposed in its datasheet.

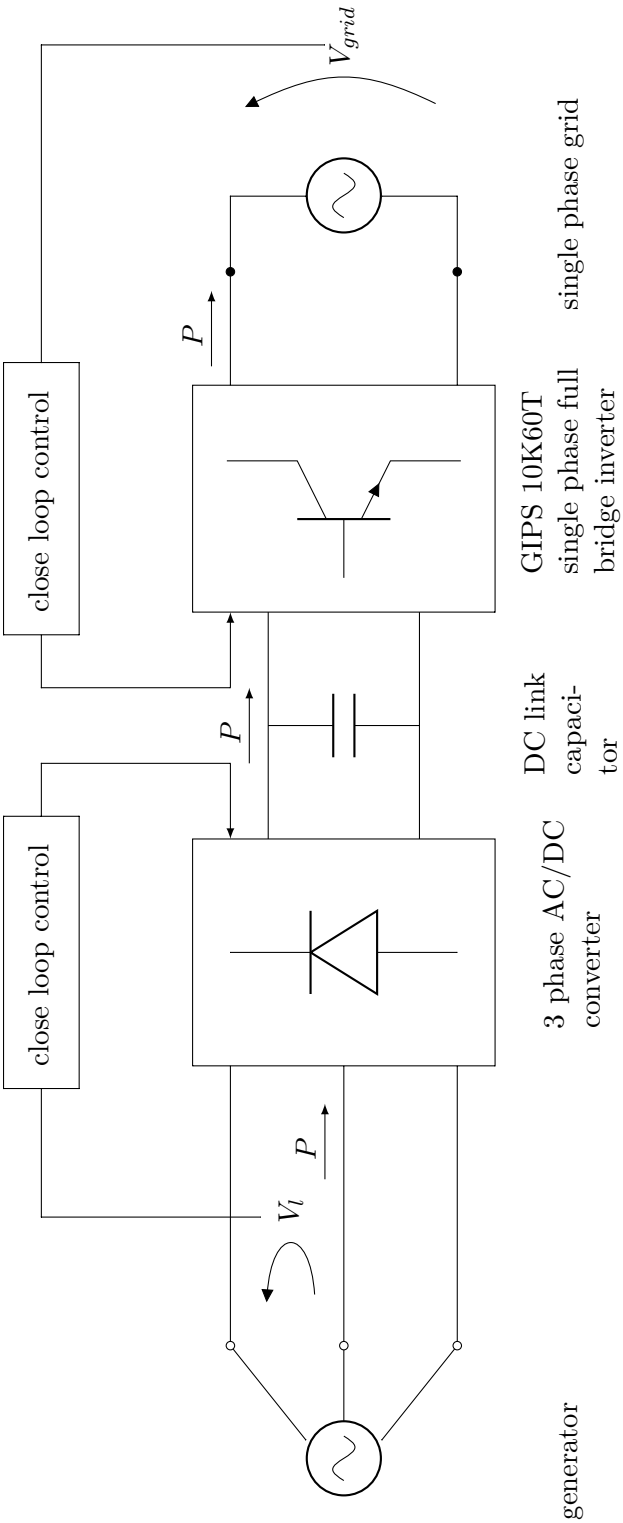


Figure 11.1: Starting point for the physical tests

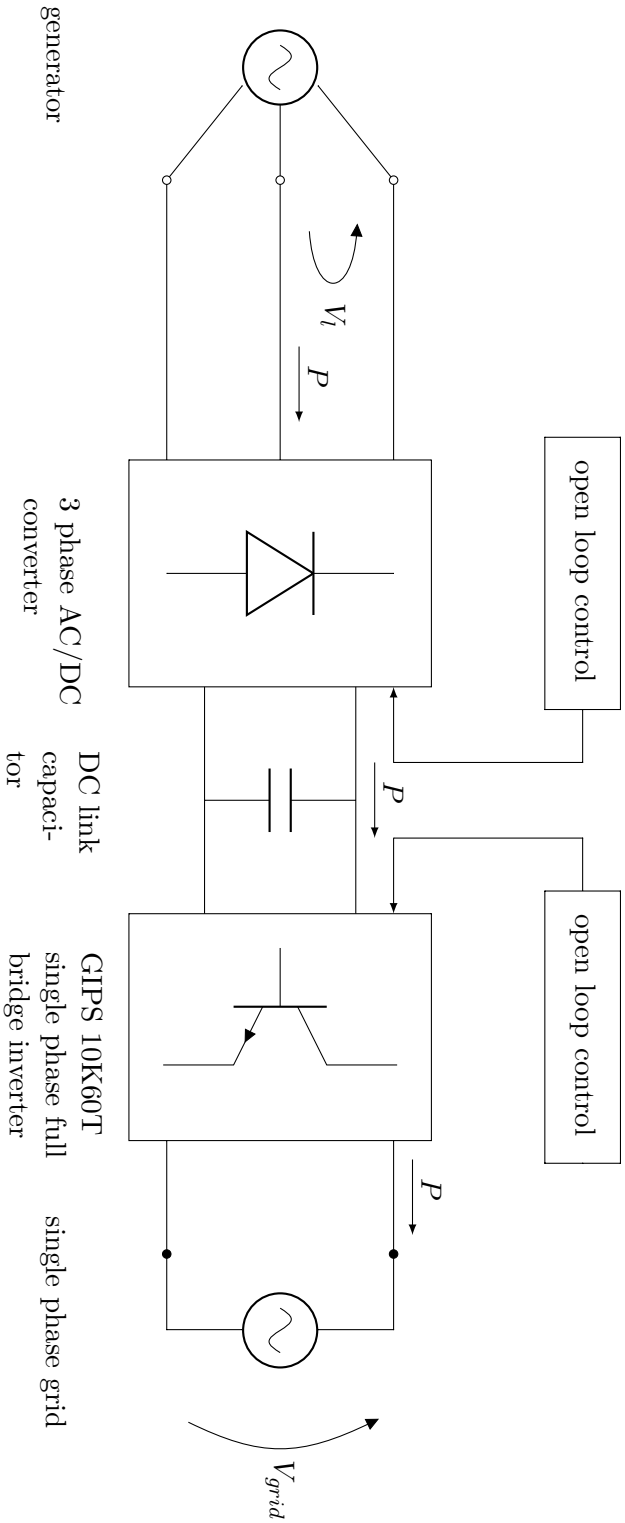


Figure 11.2: Open loop architecture

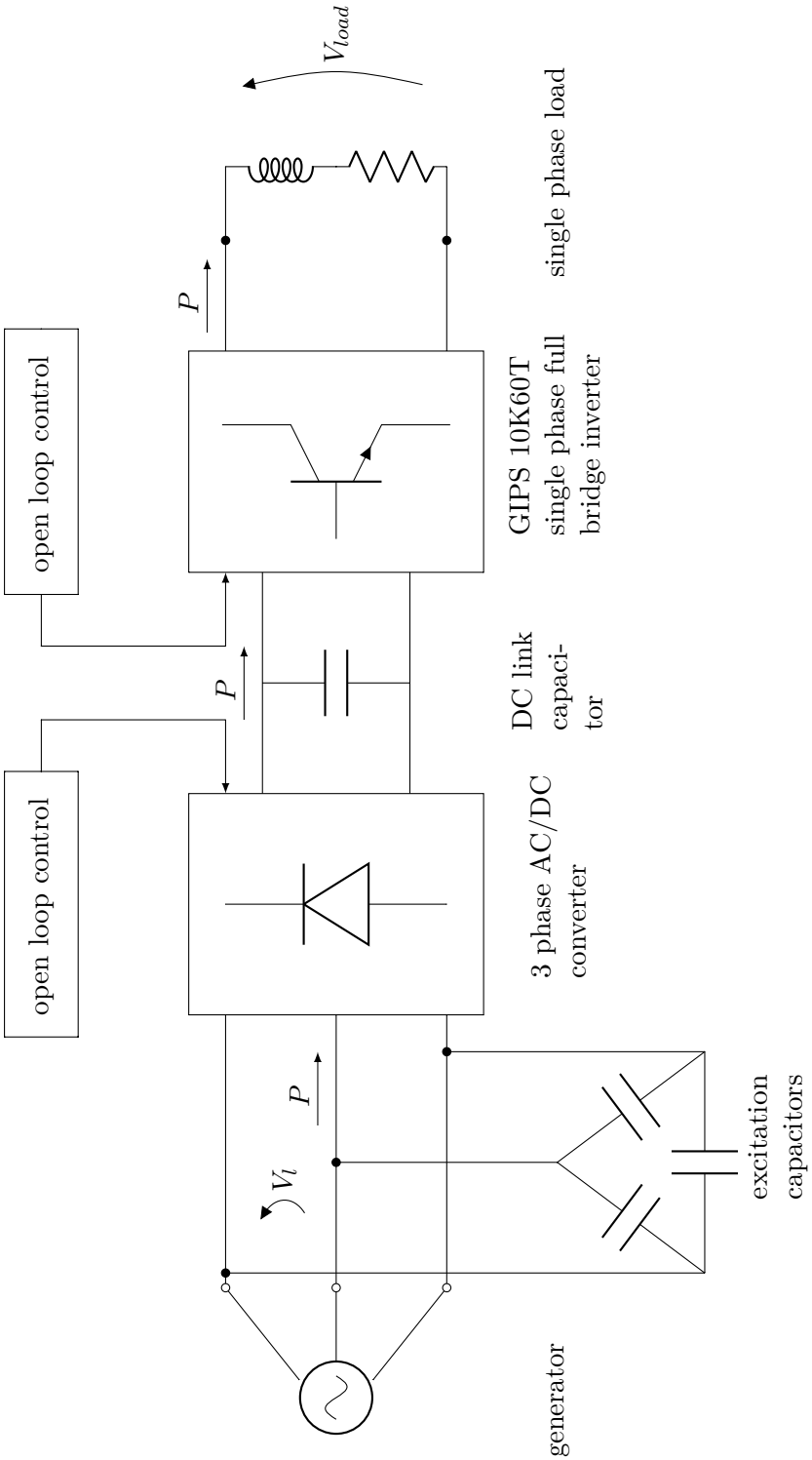


Figure 11.3: Open loop architecture with a passive load

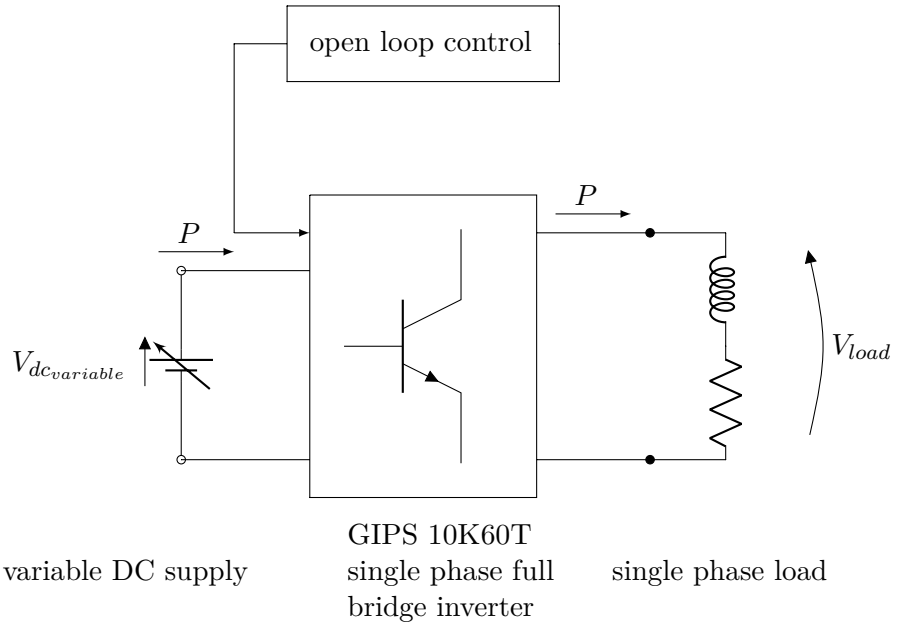


Figure 11.4: Open loop architecture with a passive load and a variable DC supply

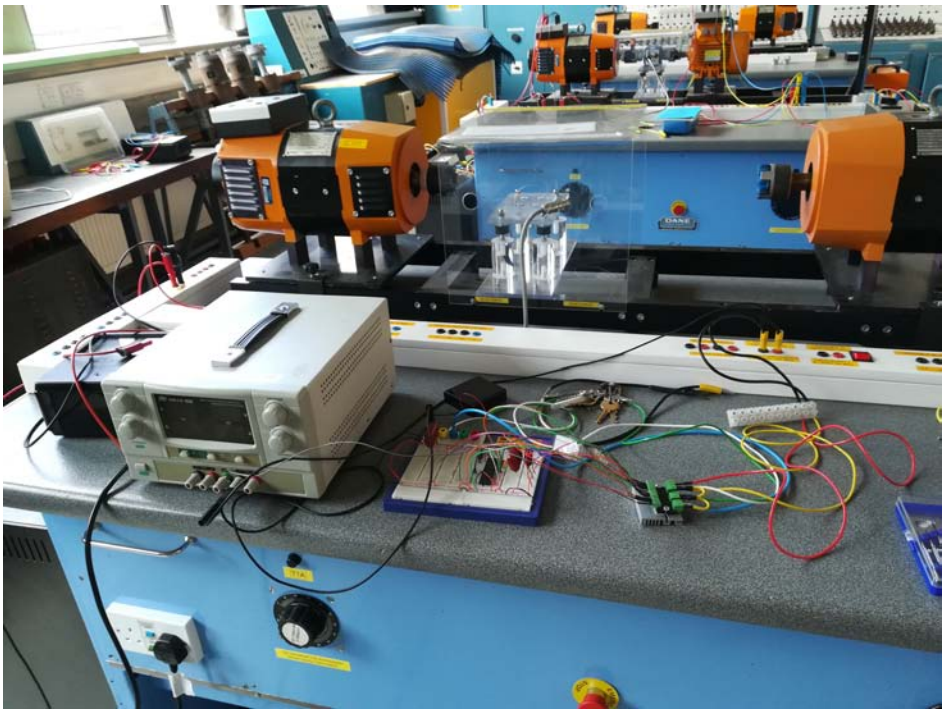


Figure 11.5: Photo of the components for both trials

11.4.2 Second trial

Extreme care this time, has been given in regards of setting the third leg switches to an open state and a further shortening of all the leads involved has been undertaken hence reducing all the possible spikes that can happen during a real test. The values used in table 11.1b has denoted now a reliability of the inverter up to the maximum voltage of our DC variable source. From the datasheet, we theoretically can have pushed the inverter for $300V$ more and with higher currents as well. Another advise was to ground the third leg for safety reasons not having floating points anymore. Figure 11.5 presents a picture of the physical circuit and components used during the entire project here implemented altogether. This might be helpful for future works.

12

Conclusions and future works

During the realisation, many fields of engineering has been employed such as a great knowledge of microgrid implementation and programming. Almost 6 months of work was done and some considerations has naturally been drawn. It is first to be said that the self-excited induction generator matched with the microcontroller in addition with a single phase AC load is a complex system hence some numerical results might differ from the simulations despite the fact that the theoretical laws are abided anywhere in the system.

12.1 Conlusions

The lab experiments proved that the connection of capacitors through the induction generator windings along with applying mechanical energy to the shaft allows the generation of electrical power (chapter 2 and chapter 3). Afterwards, the simulation shows that the behaviour of the SEIG is similar to what is known in literature. The MATLAB/SIMULINK control architecture is completely satisfying showing that a wide option of different grid features can now be analised correctly, as well as the good path undertaken by programming the dsPIC along with the breadboard usage for interfacing the inverter in order to create a sinusoidal output. A consistent part of the time has been taken to properly tuning the PI controllers even though some calculations, thanks to the whole transfer function, helped as a first orientation for proportional and integral gain values. Particular care has also been given to the inverter since an economic one was adopted (worthing $\simeq 16$ €) and it played a fundamental role affecting the success of the entire report because of the IGBT delicacy.

12.2 Future works

Having the overall circuit working in the best way possible, the physical model in future can be implemented with a more expensive and performing inverter reaching thus higher voltages and increasing the reliability of the micropower plant. In addition, another project could take into account the truthfulness of the here mentioned MATLAB/SIMULINK model, studying up to what values of time constants this acts as a stable approach or viewing what range of saturation parameters can maintain the simulation outside the unstable region. Another future work would be represented by the connection of the several components here employed to the real-life working conditions testing them and see how they behaves when a sudden or gradual change might appear. Finally, an upgrade is to physically connect everything as in the final simulation mentioned in section 7.8 and thus adding SEIG, rectifier and close loops that bring more time constant, shortening the range where the system is considered stable.

Bibliography

[1] Godoy, Simoes Martin, & Farret Felix (2005), *Alternative energy systems: design and analysis with the induction generator*, CRC press.

[2] Chapman, J. Stephen (1985), *Electric machinery and power systems: fundamentals*, Mc Grew Hill.

[3] Rodrguez, Vilar Lucia (2011), *Design of a control system with an induction generator for an isolated micro-hydro power plant*, Dublin Institute of Technology.

[4] Marciniak, Mariusz (2013), *Small hydropower system interface to the grid*, Dublin Institute of Technology.

[5] Kernighan, W. Brian, & Ritchie M. Dennis (1988), *The C programming language*, Prentice Hall. (Citato a pagina 61)

[6] Microchip (2005), *dsPIC30F Family Reference Manual*, Microchip Technology inc. (Citato alle pagine 68 e 76)

[7] Microchip (2005) *dsPIC30F4011/4012 Data Sheet*, Microchip Technology inc. (Citato a pagina 64)

[8] Microchip (2005), *dsPIC30F Family Reference Manual*, Microchip Technology inc. (Citato alle pagine 68 e 76)

[9] ST (2011), *STGIPS10K60T Data sheet*, ST microelectronics.

[10] ST (2012), *3-phase Motor control demonstration board featuring IGBT intelligent power module STGIPS10K60A*, ST microelectronics.

[11] M. Aoulkadi, & A. Binder, & G. Joksimovic (2005), *Additional losses in high-speed induction machine - removed rotor test*, University of Dresden.

[12] Yihua Hu, & Yang Du2, & Weidong Xiao, & Stephen Finney, & Wenping Cao (2015), *DC-link Voltage control strategy for reducing capacitance and total harmonic distortion in single-phase grid - connected photovoltaic inverters*, The Institution of Engineering and Technology.

[13] Minh-Khai Nguyen, & Geum-Bae Cho, & Youn-Ok Choi, & Myoung-Han Yoo, & Tan-Tai Tran (2012), *DC-Link Voltage control in single-phase switched-boost inverter*, IEEE pubblication.

[14] Scutaru, Gheorghe, & Apostoaia Constantin (2012), *MATLAB-Simulink model of a stand-alone induction generator*, IEEE pubblication.

[15] Xiangdong Sun, & Yongni Liu, & Biying Ren, & Majing Yu (2016), *Research on control strategy of a singlephase inverter with good output voltage quality*, IEEE pubblication.

[16] P. Pillay, & R. G. Harley, & E. J. Odendal (1984), *A comparison between star and delta connected induction motors when supplied by current source inverter*, University of Natal (South Africa).