Politecnico di Torino

Polytechnic of Turin

Master's degree course in Mechatronic Engineering

A.y. 2022/2023

28th October 2022

Design of a Hardware-In-the-Loop (HIL) eDrive system based on FPGA and microprocessor

Supervisors:

Prof. Radu Bojoi

Ing. Antonio Vitale

Candidate:

Andreas Jonsson

Dedication

I wish to thank a lot prof. Radu Bojoi and Eng. Antonio Vitale who, with their great patience and availability, helped and supported me in carrying out this work, guide and source of valuable advice during the whole process of drafting the thesis.

My heartfelt thanks go to all the professors of my master's degree course in Mechatronic Engineering who, in these two years of universities with their availability and cordiality, accompanied me in my career path together with all the staff of the Polytechnic of Turin with whom I collaborated.

I deeply thank my family who have always been next to me and has never made me lack support and help, encouraging me to achieve my goals and crown my multiple dreams. Special thanks go to my dad Michael and my mom Tiziana, who always have supported my choices with valuable advice and helped in taking rational decisions, and with them a special thanks also to my irreplaceable sister Francesca.

Thanks again to my grandmothers Marisa and Birte for their support, affection and love they have always given to me, to my grandfather Henning who so much I wanted to be here in what I consider to be one of the most important days of my life.

Thanks to Luca and Fabio with whom we shared precious moments together during the thesis process in the company.

Thanks to my best friends Alessandro, Nicolò, Emanuele and Trim to support me during my whole life, in the bad and the good circumstances, and with which I shared and still share wonderful moments.

To Antonio, friend and fellow student, for our reviews before each exam.

A thank you to all the people I have not mentioned explicitly in these pages, including my childhood friends Silvia, Edoardo and Gabriele who have had, and will always have, an important role in my life: each of them is indelibly imprinted in my heart.

Ringraziamenti

Desidero ringraziare molto il mio referente accademico, l'Ing. Radu Bojoi, e il mio referente aziendale, l'Ing. Antonio Vitale, che con la loro grande pazienza, disponibilità e reperibilità, mi hanno aiutato e supportato nello svolgimento di questo lavoro, guida e fonte di preziosi consigli durante tutto il percorso di stesura della Tesi.

Un grazie di cuore va a tutti i professori del mio master in Ingegneria Meccatronica che, in questi due anni di università con la loro disponibilità e cordialità, mi ha accompagnato nel mio percorso professionale insieme a tutto lo staff del Politecnico di Torino con il quale ho collaborato.

Ringrazio profondamente la mia famiglia che mi è sempre stata accanto e non mi ha mai fatto mancare sostegno ed aiuto, spronandomi a raggiungere i miei intenti e a coronare i miei molteplici sogni. Un grazie speciale va a papà Michael e a mamma Tiziana, che hanno sempre indirizzato le mie scelte con preziosi consigli e aiutata nel prendere razionali decisioni, ed insieme a loro un grazie speciale anche alla mia insostituibile sorella Francesca.

Grazie ancora alle nonne Marisa e Birte per il sostegno, l'affetto e l'amore che mi hanno sempre dato, a nonno Henning che tanto avrei voluto assistesse a questo che considero uno dei giorni più importanti della mia vita.

Grazie a Luca e Fabio con cui abbiamo condiviso momenti preziosi durante il processo di tesi in azienda

Grazie ai miei migliori amici Alessandro, Nicolò, Emanuele e Trim a sostenermi durante tutta la mia vita, nel male e nelle buone circostanze, e con cui ho condiviso e ancora condividere momenti meravigliosi

Ad Antonio, amico e compagno di studi, per i nostri ripassi prima di ogni esame.

Un ringraziamento lo dedico a tutte le persone che non ho nominato esplicitamente in queste pagine, tra cui i miei amici d'infanzia Gabriele, Edoardo e Silvia che hanno avuto, hanno e avranno sempre, un ruolo importante nella mia vita: ognuno di loro è impresso in maniera indelebile nel mio cuore.

INDEX

Abstract

Nowadays Hardware-in-the-loop simulation (HIL) techniques are increasingly in demand since, in addition to ensure excellent efficiency, they are increasingly used as they are able to allow simulations that would normally damage or destroy the machine.
In this way, test beyond the normal range of operation can then detect whether the control system can operate the machine safely.

The project carried out at Kineton S.r.l. involves the implementation of an electric motor model, a permanent magnet synchronous motor (PMSM), in a hardware-in-the-loop environment, using an external control unit with sensorless FOC control as a Device Under Test.

The implementation of the model will be partly on the processor and partly, given the high frequency band, will be done using the FPGA.

The project has as its application an electric turbocharger, for this reason as an additional objective is the achievement of high speeds and performance.

References:

1. *https://sites.google.com/site/storiaindustriaauto/evoluzione-dell-industria-automobilistica-nella-societa-e-nell-economia;*
2. *slides 01 - Casetti - networking for connected vehicles;*
3. *https://www.speedgoat.com/solutions/testing-workflows/hardware-in-the-loop;*
4. *https://www.flexsci.com/productSolutions/sys/hil.htm;*
5. *https://farelettronica.it/motore-elettrico-tipi-funzionamento/;*
6. *https://scienzapertutti.infn.it/chiedi-allesperto/tutte-le-risposte/553-48-motori-sincroni-e-asincroni-quali-i-migliori];*
7. *https://it.farnell.com/motor-control-permanent-magnet-sync-motor-pmsm-technology;*
8. *Bojoi - Electrical technologies for e-Mobility;*
9. *https://www.electrical4u.com/control-of-electrical-drives/;*
10. *https://www.electronics-tutorials.ws/systems/closed-loop-system.html;*
11. *https://it.mathworks.com/help/mcb/gs/implement-motor-speed-control- by-using-field-oriented-control-foc.html;*
12. *https://www.ideegreen.it/come-funziona-inverter-49215.html;*
13. *https://imperix.com/doc/help/xilinx-system-generator;*

List of Figures:

Figure 2.18 -https://it.mathworks.com/solutions/power-electronics-control/clarke-and-park-transforms/_jcr_content/mainParsys/band_copy_1227855798/mainParsys/columns_1606542234_c/1/animation_copy.animation.mp4/1614705044648.mp4;
Figure 2.19 -https://it.mathworks.com/solutions/power-electronics-control/clarke-and-park-transforms/_jcr_content/mainParsys/band_copy_1227855798/mainParsys/columns_1606542234_c/1/animation_copy_copy.animation.mp4/1614705044671.mp4;
Figure 2.20 - Lesson 06-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.21 - Lesson 08-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.22 - Lesson 08-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.23 - Lesson 08-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.33 - https://control.com/uploads/articles/closed-loop_motor-control_intro_figure3_1.jpg;
Figure 2.34 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.35 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.36 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.37 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.38 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.39 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.40 - https://electronicscoach.com/wp-content/uploads/2019/11/open-loop-control-system-1.jpg;
Figure 2.41 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.42 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.43 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.44 - Lesson 09-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.45 - https://www.omron-ap.com/service_support/FAQ/FAQ00549/FAQ00549-2.jpg;
Figure 2.46 - https://it.mathworks.com/help/mcb/gs/pmsm_foc.png;
Figure 2.47 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.48 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.49 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.50 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.51 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.52 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.53 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.54 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.55 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;
Figure 2.56 - Lesson 05-Bojoi -Electrical Technologies for e-Mobility;

Figure 2.57 - https://upload.wikimedia.org/wikipedia/commons/thumb/7/76/Turbocharger.jpg/1280px-Turbocharger.jpg
Figure 3.1 - https://upload.wikimedia.org/wikipedia/commons/thumb/2/21/Matlab_Logo.png/667px-Matlab_Logo.png;
Figure 3.2 - https://it.mathworks.com/help/simulink/slref/tool_simulink_editor.png
Figure 3.3 - https://xilinx--c.documentforce.com/servlet/servlet.ImageServer?id=0152E000003pLGO&oid=00D2E000000nHq7;
Figure 3.4 - https://www.dspace.com/_clickr/cfml/index.cfm?src=/shared/img/2018/ConfigurationDesk_1400x788px_180327.png&width=787;
Figure 3.5 - https://d2368tcediwknr.cloudfront.net/video/products/ControlDesk/dSPACE-ControlDesk03-Measurement_1920x1080_2017-11_English.png;
Figure 4.8 - https://www.dspace.com/_clickr/cfml/index.cfm?src=../../../../../../../files/jpg20/scalexio_processing_unit_700x394px_1711301.jpg&width=300&height=170;
Figure 4.10 - https://www.dspace.com/_clickr/cfml/index.cfm?src=../../../../../../../files/jpg11/SCALEXIO_System_DS2680_1400x788px_01_150417.jpg&width=300&height=170;
Figure 4.13 - https://www.dspace.com/_clickr/cfml/index.cfm?src=../../../../../../../files/jpg27/ds6602-board_1920x1080px_200812.jpg&width=300&height=170;

1. PREFACE

1.1 Aim of the thesis project

The aim of the thesis project, carried out at the company Kineton S.r.l., involves the design of a Hardware-in-the-loop (HIL) system, low voltage, consisting of an external control unit with sensorless FOC control (and CAN communication), that will perform the function of Device Under Test (DUT); a model of a permanent magnet synchronous motor developed on a processor and an inverter model (together with the Clarke & Park transformers) developed on FPGA, given the high frequency band.
In this thesis project, the permanent magnet synchronous motor has an electric turbocharger as application, so that the additional objective is to achieve high speed and performance of the motor itself.
A phase of study and research of all theoretical topics related to the project will be deepened in the first chapters in order to achieve the goal. Then we will discuss in detail the project carried out with the exposure of the results obtained.

1.2 Structure and characteristic of the company Kineton

Kineton was founded in 2017 as an innovative startup by a group of ten professionals specializing in automation and software development. Kineton S.r.l. is an SME that offers cutting-edge engineering services.
A few years after its foundation it has about 400 employees located in Naples, Milan, Reggio Emilia, Turin, Birkenhead (GB) and Tirana (AL).

2. INTRODUCTION

2.1 HIL: hardware in-the-loop structure

In order to replace the traditional testing method that might lead to time-consuming, expensive and potentially unsafe algorithms, test engineers have substituted this method with Hardware-in-the-loop (HIL) testing. This type of simulation allows testing of the system in its intended modes of operation as well as under dangerous or emergency situations without risking loss of life or valuable assets.
Moreover, critical or strange environmental conditions that may be almost impossible for system test, like outer space or icy roads in summer, can be simulated using computers and appropriate software algorithms.
Hardware-in-the-loop simulations run at real time speed and perform I/O with the system or subsystems under test, in this way the test item believes it is operating as a component of a real systems in its operational environment.
These systems or subsystems can be tested both under nominal conditions or under their intended operational boundaries in order to prove their proper operation under all conditions.
The ability to fully test subsystems using simulation early in the development process can significantly reduce the debugging time and project risk compared to the different approach of waiting until a prototype is completed before performing integration and test.

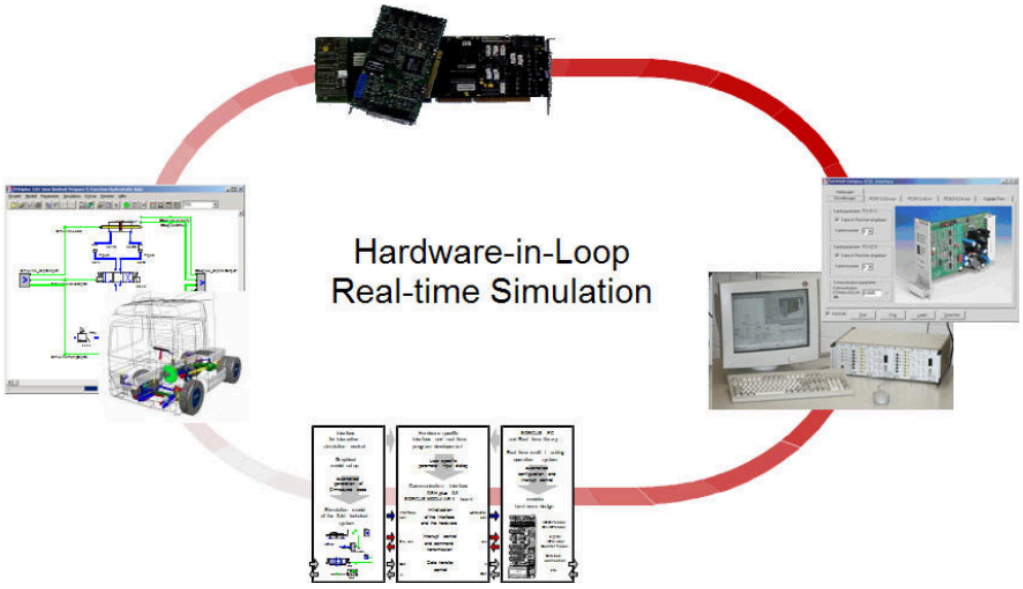


Figure 2.1 - Hardware-the-loop Simulation

Some advantages of using and Hardware-in-the-loop (HIL) simulations are:

- Improve safety: HIL permits to test or simulate dangerous situations without both risking damage to the vehicle and the safety of the person who normally perform test simulation;
- Increases the quality: if used in a Model-based design process it will grow together with the real system project and will be able to be continuously used for testing control system.
- Save time: error cost grows with the time spend to find them. Usually, the development of the machines and the control system takes place in parallel, these lead that control system errors are found during the commission. With the Hardware-in-the-loop (HIL) testing errors can be found and fixed first, I order to reduce the commission time;
- Save money: testing a real system could lead to high cost because it can require expensive prototypes, high startup costs or expensive safety precautions. HIL can avoid all these types of costs. It requires lot of time in the implementation phase;
- Human Factor: it can be used for testing human interaction on the machines, this permit to test if they can be used easily and comfortably;

The design of the HIL is separated into the design of the electrical aspects of the simulator, the modeling of the I / O and the system, and finally the creation / execution of the tests.
The hardware used for the project is SCALEXIO from *dSpace*.

Figure 2.2 - Scalexio

This hardware is composed by:

- Real-time processor: is based on an industrial PC with an Intel Core i7 quad-core processor, a real-time operating system (RTOS) and a PCIe plug-on card designed by dSpace for communication between I / O and real-time processor. It is connected to the host pc via Gigabit Ethernet, used to configure the entire simulator, load real-time applications and monitor / control the HIL simulation;
- I/O network: real-time processor and I / O boards communication has an effect on the minimum cycle time of the simulation step. dSPACE has developed a new serial I / O network called IOCNET, used for the first time in SCALEXIO.
- I/O boards: HighFlex, MultiCompacte FPGA I/O.

SCALEXIO is configured through the ConfigurationDesk software. The process is divided into:

- Describe the connected external devices;
- Select the I / O functions for each signal;
- Link the I / O functions to the plant model.

Figure 2.3 - Configuration Desk

2.2 FPGA

A Field Programmable Gate Array (or FPGA), in digital electronics, is a programmable logic device where inside it you can find logic circuits that can be reprogrammed through a specific string of configuration bits, so as to be able to implement both sequential circuits (for example flip-flops or other more complex circuits) and combiners (AND / OR gates or more complex circuits).

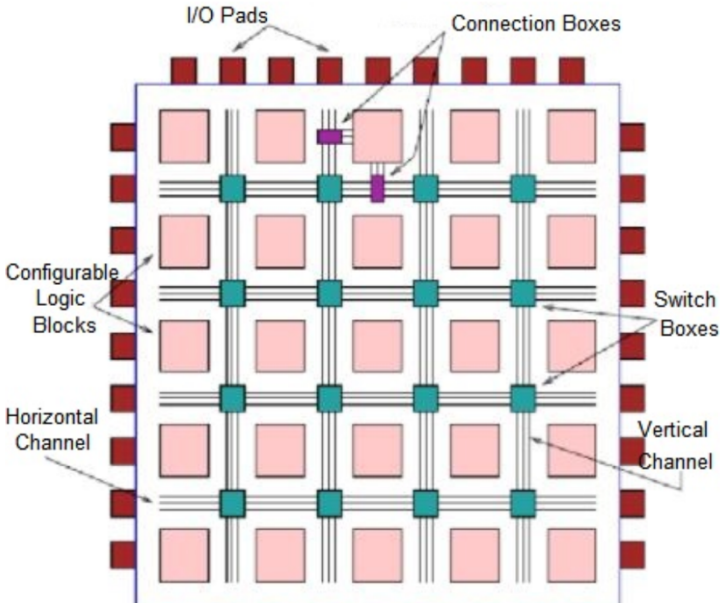


Figure 2.4 - FPGA

Programming and any changes are made using appropriate hardware description languages such as Verilog or VHDL.

The internal structure of an FPGA generally consists of a matrix of configurable logic blocks (CLBs) connected to each other through programmable interconnections. At the edges of this matrix there are the input and output blocks (I/O).

The CLBs perform the logic functions and are put into communication by a set of interconnections, while the I / O deal with interfacing the circuit with the outside world. Each of these logic blocks contains a set of elements, including a lookup table (LUT), multiplexer, and register, which can be configured to behave as required.

Many FPGAs use 4-input LUTs that can be configured to implement any 4-input logic function. The LUT output is connected directly to one of the logic blocks outputs and to one of the multiplexer inputs. The other input to the multiplexer is connected directly to an input of the logic block. The multiplexer can be configured to choose one of these inputs.

The multiplexer output feeds the register input. Each register can be configured to function as a flip-flop or latch. The clock of each register can be active high or active low. The interconnection can be configured to connect any output from any logic block to any input of other logic blocks.

The primary inputs of the FPGA can be connected to the inputs of any logic block and the outputs of any logic block can be used to drive the primary outputs of the device. Peripheral interface functions such as CAN, I2C, SPI, UART and USB can be included as hard core in the chip.

High-speed buses such as AMBA and AXI are typically used for communications between processor cores, interface functions and programmable structure.
Benefits:
Flexibility:

- It can implement any type of circuit, from a simple counter to an entire microprocessor.
- It can reconfigure the implementation during the development phase without changing the hardware.

Speed:

- The execution can be done in parallel.
- The control loop can be made with a very high sampling time.

2.2.1 Xilinx System Generator for DSP

Xilinx System Generator for DSP (SysGen) is a particular tool present in MATLAB, specifically on Simulink, for high-level design and programming to define, test and implement high performance DSP algorithms on Xilinx devices.

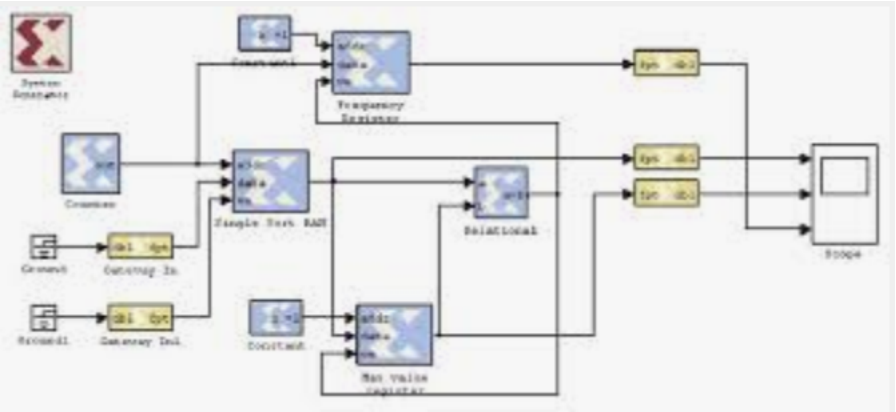


Figure 1.5 - Xilinx for DSP block

Digital signal processing algorithm (DSP) is a specific step-by-step procedure for mathematical calculations designed to manipulate digital signals.
It offers a limited set of features compared to Simulink.
It needs the System Generator block to specify the hardware target, clock, timings and configuration parameters.
The key features are:

- DSP algorithms.
- It supports fixed and floating operations.
- Automatic generation of HDL.
- Post synthesis and post implementation temporal analysis.

2.2.2 RTI FPGA Programming block

RTI FPGA Programming block is a simulink blockset to integrate an FPGA application into a dSPACE system. It is used to implement:
the interface between FPGA and I / O channels and the interface between FPGA and real-time processor.

Figure 2.6 – RTI programming block

The execution time of an FPGA is faster than the execution time of the real time processor.

- Develop two subsystems in Simulink: processor application and FPGA model;
- Implements the functions of the FPGA (XSG Blocks);
- Set the FPGA interface (FPGA_SETUP_BL);
- Simulate the behavior of the FPGA subset;
- Set the communication model of the processor (goto, from)
- Implement features from the processor side;
- Simulates the entire model;
- Build the FPGA application from the FPGA submodel;
- Export the FPGA application and processor model to ConfigurationDesk;
- Configure the signal properties;
- Build the processor application using ConfigurationDesk;
- Run the entire model in real-time hardware.

2.3 Introduction to automotive

Since its inception, automotive industry has been in a constant evolution, between the ups and downs of global or territorial crises, and it has radically affected the economies of most countries, as well as society itself.

Figure 2.7 - Automotive

In the first decades of the 900s, in a country that has the typical characteristic of a growing state like United State of America, first automotive industries are born.

In Europe, automotive industry was already there since the last decade of 800s, obviously with models based on carriages.

With the 900s they began to be produced passenger cars much more practical and versatile but reserved only to the “elite” category of the population. Due to the outbreak of the World War I, automotive industries suffer a major arrest, mostly in Europe.

From the first happenings of the twentieth century you can guess the huge evolution of the automotive industry, above all we can see how it was able to adapt to both historical conditions and to condition historical

events in turn, as in the case of the World War I, where motor vehicles brought a huge change in warfare strategy.

Since its beginning vehicle like passenger car have been dominated by mechanical component. Electrical parts played a peripheral role, mainly for: ignition, starter, Battery charging, user information or illumination. Mechanical part instead, were responsible for torque generation, distribution and for controlling dynamic vehicle behavior.

With the passing years the need to improve performance and new features grew significantly. The limit of what was possible using only mechanical parts was reached. Consequentially, electrical and electronic parts were massively introduced to overcome this limit.

Nowadays, the electronic component is ruling on the mechanical parts: more than the half of the cost of developing a vehicle comes from the design and validation of the electrical part.

A modern vehicle is an interconnection of several intelligent entities cooperating towards a common goal in order to achieve some performances objective like, for example, lower emission, longer ranger or higher safety.
Things changed “a little bit” since 1965, more than a hundred ECUs embedded, connected with several in-vehicle networks, for systems/services have different requirements and require different technologies.
In-vehicle network introduce several advantages with respect of using discrete wiring:

- Reduce the amount of wiring: this mean less weight, cost and wasted space and more eco-friendliness;
- Improve Flexibility: more software-controlled functions, less hardwired functions, more freedom in components placement and wire routing;
- Share information more easily: in vehicle networks can broadcast information to al interested parties; Gateways can propagate information from one network to another and even more importantly, they can make it available to other vehicles;

In a vehicle we can classify different type of ECUs:

- Powertrain and Chassis Control: Engine, automatic transmission, hybrid control, steering, brake suspension and so on;
- Body Electronics: Instruments panel, key, door, window, lighting, air bag, seat etc.;
- Multimedia Applications: Car audio, car navigation, traffic information, Electronic toll collection (ETC), back guide monitors etc.;
- Integrated Systems/Services: Electronic stability control, pre-crash safety, Parking assistance, lane keeping assistance, autonomous driving etc.;

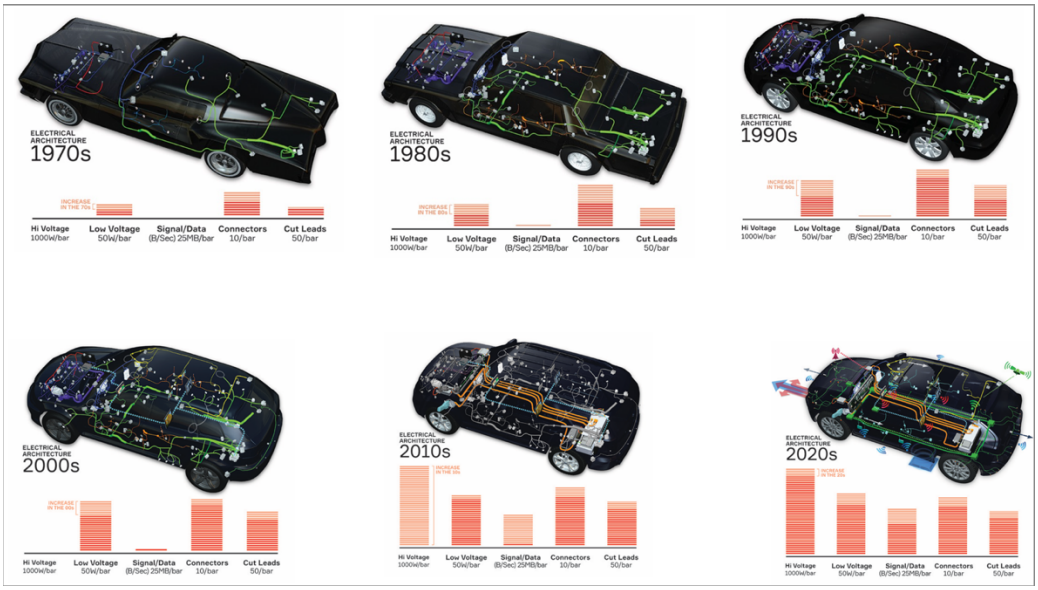


Figure 2.8 – Vehicle evolution

2.4 Electric motors

The internal combustion engine is destined to disappear to make room for new solutions like electrical motors and batteries.
The electric motor constitutes a system that convert electric energy into mechanical energy for movement. There are different typology exists of electric motor and they can be used for this purpose. An electric motor must be able to:

- Starting from a null velocity develop a significative torque;
- ensure significative peak power in order to have the same performance as traditional vehicles;
- have an electric motor piloting and control system as simple as possible;
- be compact and light;
- cost relatively little;
- have high efficiency at the highest levels;
- working as a generator when slowing the vehicle.

Synthesizing, an ideal motor for traction application must have excellent characteristics like high torque, high power density and good energy efficiency.

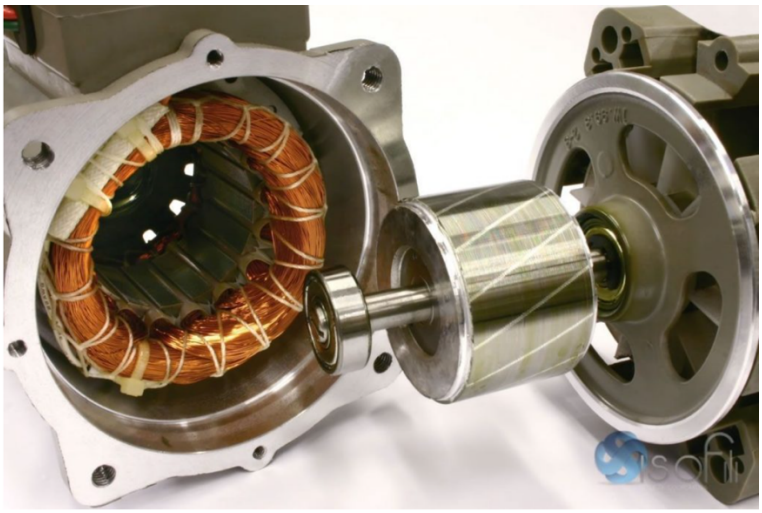


Figure 2.9 - Electric motor

Typology of electric motor:

- *DC motors*: DC motors should be mentioned in this sequence as they were the first motors to be widely used in traction applications in the early part of the last century. They offered, and still offer, concrete positive aspects: high starting torque, the ability to support sudden increases in load, easy speed control, construction simplicity and low costs.

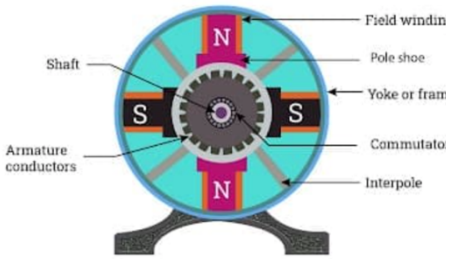


Figure 2.10 - DC Motor

But they have a major drawback: the need for electrical contacts, brushes, on a rotating part to make it possible to switch polarity on the rotor windings.

This aspect makes them need constant maintenance, due to the consumption of brushes normally made of carbon, and this has practically put them out of the market for these types of applications;

- *Permanent Magnet Synchronous Motors (PMSM):* Permanent magnet synchronous or sinusoidal brushless motors are essentially intended for high performance drives where the particular specifications justify their cost, which is usually high due to the presence of valuable permanent magnets in the mobile element. The electromechanical conversion that they carry out follows the operating principle of electrodynamic systems in which, however, the conductors on which the forces act is placed in the fixed part (stator) and the rotor is set in motion due to the physical reaction principle.

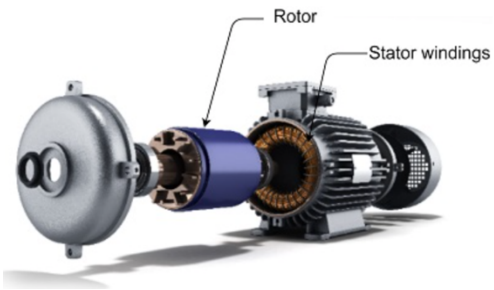


Figure 2.11 - PMSM

- *Continuous brushless motors (BLDC):* they do not suffer of the inconvenience need for maintenance of which brush motors suffer from. They have similar characteristics to those in DC: a great couple at start-up, high energy efficiency and they can be designed to achieve one power density particularly compact dimensions.

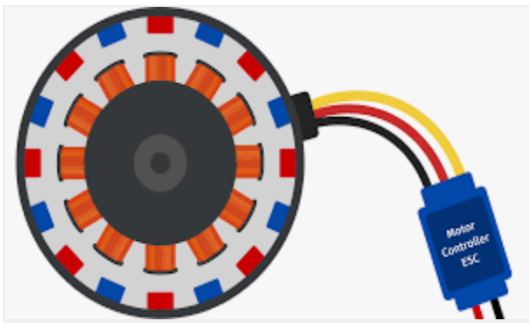


Figure 2.12 - Brushless motor

For these characteristics they are among the preferred types of engine in e-bike and two-wheeled medium power applications, where the compactness factor is fundamental.

In BLDC motors the windings are not on the rotor but on the stator, the fixed part, while it is the rotor that contains the permanent magnets.
Since the windings are fixed, the brush-rotating commutator combination is not necessary, as the rotation is obtained by controlling the magnetic field generated by the windings that make up the rotor while the magnetic field that is generated by the magnets that make up the stator is fixed. To change the rotation speed, the voltage must be changed;

- *Commutated reluctance motor:* it consists of a stator with salient poles, individually linked with excitation windings and a rotor with teeth, in different numbers from that of the stator poles.

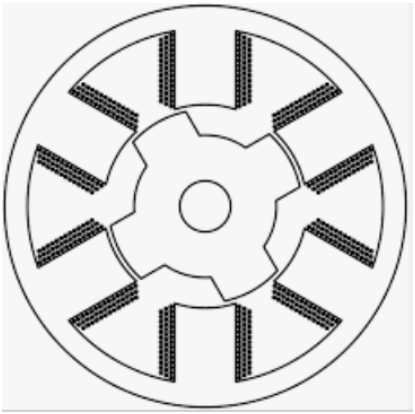


Figure 2.13 - Reluctance Motor

The couple is produced by the tendency of the rotor teeth to align with the
stator poles, whose windings are cyclically powered by the electronic converter.
The switched power supply circuit, compared to the synchronous reluctance machine converter, is somewhat simpler, as it is based on one-way switches.
The switching points must be precisely established, and the waveforms of the currents and the shape of the pole pieces must be designed
suitably, to minimize the torque ripple and the noise, which intrinsically can be generated as a consequence of the power supply discontinuities of the single poles;

- *Induction motors (IM):* is an alternating current electric motor in which the angular velocity of the rotor shaft is lower than the rotation speed of the magnetic field generated by the windings of the stator.

Figure 2.14 - Induction motor

This engine has a fixed part
said stator and a movable part called
rotor. The stator has some
internal grooves that house the conductors of the stator winding which can be three-phase or two-phase.
The rotor is in turn located inside the stator and has external grooves (rotor slots) to accommodate the rotor winding. In the asynchronous motor, between stator and rotor there is a small space called air gap, to allow free rotation of the rotor.

This thin air thickness measures a few tenths of a millimeter and the stator windings are generally made of resin globes, for protective purposes.
The stator generally contains an even number of windings as there are two for each power supply phase.
The stator winding is powered with an alternating current. With the arrangement of the polar pairs, out of phase with each other, the current generates an overall magnetic field that rotates in space with the same frequency as the supply current and which is called the stator field.
The rotor winding is immersed in this rotating magnetic field. The rotor turns slower than the stator field, so the magnetic flux subtended by the rotor winding changes; thus, the rotating magnetic field has currents in the rotor by magnetic induction.
These induced currents, in turn, generate a rotor magnetic field, which opposes flux variations. According to Lenz's law, the magnetic field induced in the rotor always has the opposite direction with respect to the static one;

Nowadays, we can distinguish to different type of motor: Synchronous motors and Asynchronous (Induction) motor (IM).

- Synchronous machines $\left(wm = \frac{2\pi f}{p}\right)$: rotor speed rigidly related to the AC frequency imposed by the inverter;

- Asynchronous machines $\left(wm \leq \frac{2\pi f}{p}\right)$: the rotor speed “slips” with respect to the AC frequency imposed by the stator.

Both the asynchronous and synchronous machine operate on alternating current and they consist of a fixed part with a cylindrical cavity called the stator and a coaxial moving part called rotor arranged inside the cavity mentioned before. In the stator of both types of motors there are electric wires according to three-phase stator windings.
The arrangement of these winding is made in such a way as to take advantage of the alternating current in order to create a rotating magnetic field.
A permanent magnet constitutes the rotor of a synchronous motor, while in the other motor type there are windings which are closed on themselves.

The stator currents, in both engines, generate a rotating magnetic field which, interacting with the rotor magnetic field, generate some forces that causes the rotor to rotate.
The frequency of the power supply is linked to the constant rotation speed of a synchronous motor, which is independent of the load (or resistant torque).
On the contrary, asynchronous motors are characterized by a rotation speed dependent of the load applied on the rotation axis. Indeed, the rotation velocity of the rotor is lower respect than that of the rotating field and therefore such rotation is not synchronism.
An alternating current generates a rotating magnetic field, due to the fact that all AC motors generally work without brushes, i.e. they do not need sliding contacts to operate.
However, in the case of a wound rotor asynchronous motor (the cage asynchronous motor is therefore excluded), the presence of brushes can be envisaged, the purpose of which is to add external resistances to the rotor winding. This is done to increase the engine torque at starting, that is, at the start of the engine. In this case, of course, the brushes are moved away from the rotating shaft immediately after the engine is running.
This allows a reduction in friction and improves the efficiency of the asynchronous motor. The use of brushes is also necessary if you want to have speed adjustments without using mechanical reducers.
In the case of synchronous motors with wound rotor (i.e. not with permanent magnets) there are two cases:

a) synchronous motor with brushes, necessary to bring the exciting direct current from the outside to the rotor windings;
b) brushless synchronous motor which through an external exciter and rectifiers is able to induce direct current in the rotor windings of the synchronous motor.

Contrary to alternating current motors, brushes are instead necessary (and not an optional as just mentioned above) on collector motors, that is, in direct current motors.
We now list some pros and cons of the two types of engines.

Synchronous motor:

- *Pros*: high power / weight ratio, reliability, low rotor inertia (allows high accelerations), heat only on the stator (as the rotor is typically permanent magnet)

- *Cons*: relatively high cost, at high temperatures (in the case of the permanent magnet motor) the magnet can demagnetize.

Asynchronous motor:

- *Pros:* low cost, no problem at high temperatures.
- *Cons*: relatively high cost, at high temperatures (in the case of the permanent magnet motor) the magnet can demagnetize.

2.5 Permanent Magnet Synchronous motor (PMSM)

In this paragraph we want to explain all the constructive and operational aspects of the engine under study, in particular it will be presented the mathematic model, the electrical and magnetic model and finally we will continue with the control techniques.

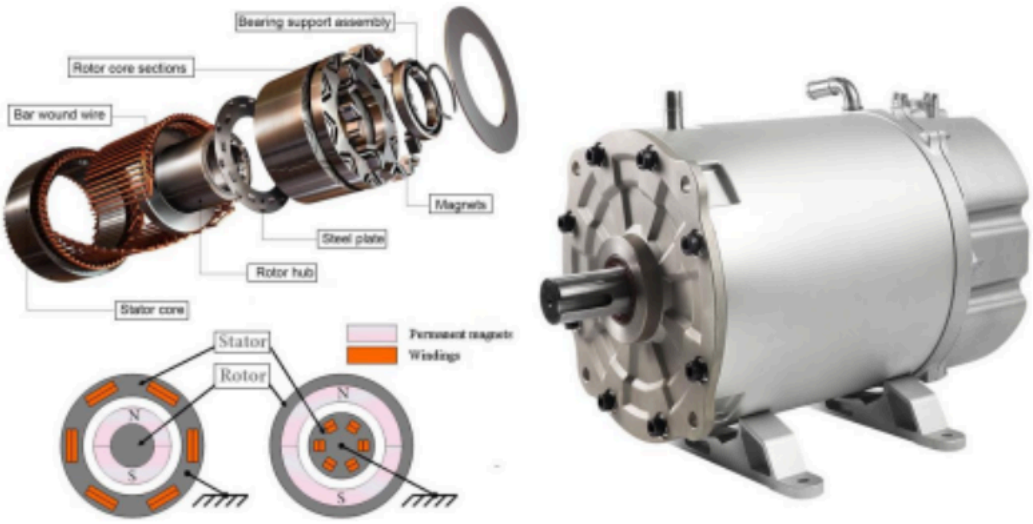


Figure 2.15 - Permanent Magnet Synchronous motor

The PMSM (permanent magnet synchronous motor) is a mix between an induction motor, due to the stator winding structure that produces a sinusoidal flux density in the transfer of the machine, and a CC motor brushless, because it is equipped with a permanent magnet rotor and stator windings.
The power density is higher than induction motors with the same ratings, since there is no stator power supply dedicated to produce the magnetic field.

Both the stator and the rotor are separated by an air gab and they have a cylindrical crown shape of laminated ferromagnetic material.
In the rotor surface we can find the permanent magnets, which generally have a differential magnetic permeability almost equal to that of air.
Rotor structures can be obtained isotropic or anisotropic from the magnetic point of view, depending on their arrangement and the shape of the rotor:

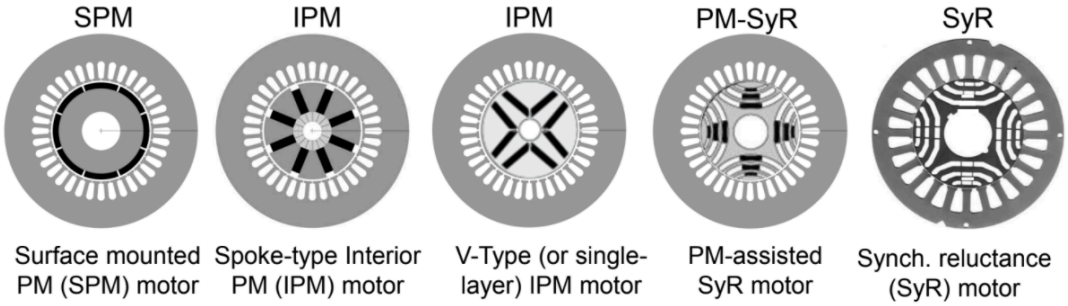


Figure 2.16 - PMSM rotor structure

The stator winding is a the three-phase type. The three phases are mutually out of phase in the space of $\frac{2}{3}\pi$ and each one is connected to a pair of terminals through which it is possible to supply them with a power supply from an external three-phase source.
PMSMs motors are characterized by a higher torque / inertia ratio and a better dynamic response due to the presence of the permanent magnet instead of a winding in the rotor which increases the flux density in the air gap. They are also more efficient and easier to cool as losses related to rotor current are eliminated. Furthermore, their power-to-weight ratio is higher than induction motors and the control strategy is easier to implement as no slip speed calculation is required.
However, the disadvantages of these motors are the variation of their properties over time and temperatures, and in order to function correctly they need high resolution position sensors, power inverters and a microprocessor to manage all the necessary components.
Another limitation is the excessive cost for high-power applications, which is the reason why their use is usually limited to a few kilowatts. All the features previously highlighted make these motors suitable for many applications, such as robotics and aerospace actuators, electric vehicles, etc., but also to different fields such as papermaking, packaging, textiles, ceramics, glass and woodworking.
Finally, the main characteristics of PMSMs can be summarized as follows:

- high flux density at the air gap;
- high power / weight ratio;

- large torque / inertia ratio, which guarantees high accelerations;
- small torque fluctuations even at very low speeds, which allows to obtain considerable accuracy in positioning;
- wide range of speed variation;
- possibility of operation at high torques, which allows to obtain rapid accelerations and decelerations;
- high efficiency and high-power factor;
- compact structure.

2.5.1 Clarke & Park transform:

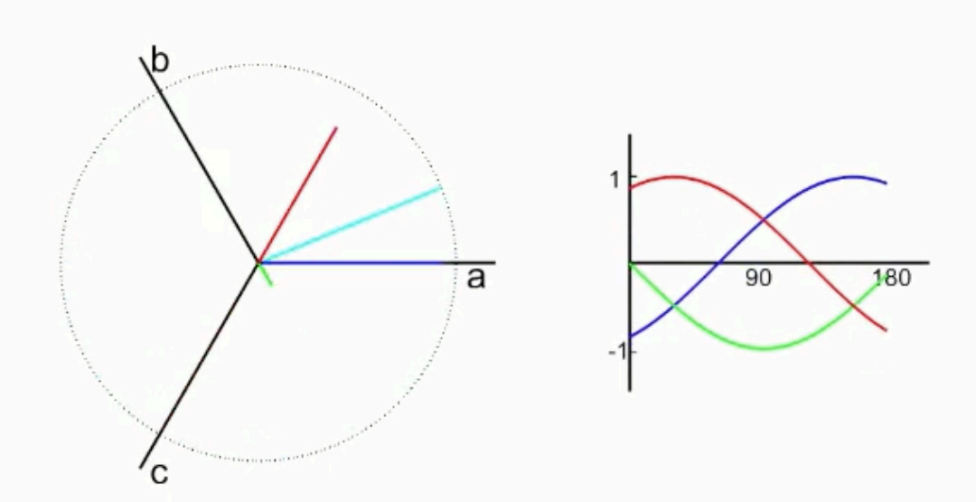


Figure 2.17 - Clarke & Park transform

Clarke and Park transforms are commonly used for field-oriented control of three-phase AC machines.

- *Clarke*:

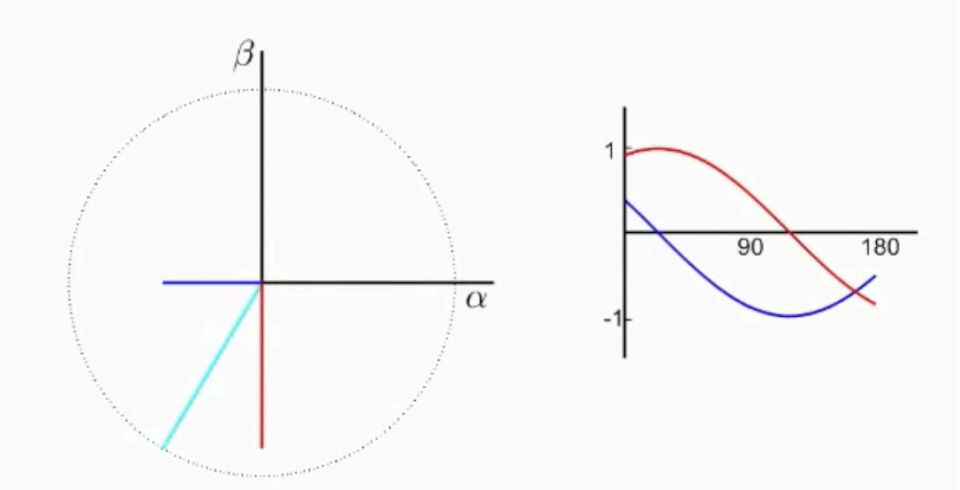


Figure 2.18 - Clarke transform

The Clarke transform converts the time domain components of a three-phase system (in frame abc) into two components in an orthogonal stationary frame (αβ).

$$\bar{x} = \frac{2}{3}(x_a + \bar{a}x_b + \bar{a}^2 x_c) \rightarrow \begin{cases} x_\alpha = Re(\bar{x}) = \frac{2}{3}\left(x_a - \frac{1}{2}x_b - \frac{1}{2}x_c\right) \\ x_\beta = Re(\bar{x}) = \frac{2}{3}\left(\frac{\sqrt{3}}{2}x_b - \frac{\sqrt{3}}{2}x_c\right) \\ x_0 = \frac{1}{3}(x_a + x_b + x_c) \end{cases} \tag{2.1}$$

In a compact form, we can write the Clarke transformation $(a, b, c) \rightarrow (\alpha, \beta, o)$ as:

$$\begin{bmatrix} x_\alpha \\ x_\beta \\ x_o \end{bmatrix} = [T] \cdot \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \quad \leftrightarrow \quad [T] = \frac{2}{3} \cdot \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{2.2}$$

and its inverse $(\alpha, \beta, o) \rightarrow (a, b, c)$:

$$\begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} = [T]^{-1} \cdot \begin{bmatrix} x_\alpha \\ x_\beta \\ x_0 \end{bmatrix} \quad \leftrightarrow \quad [T]^{-1} = \frac{2}{3} \cdot \begin{bmatrix} 1 & 0 & 1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & 1 \end{bmatrix} \tag{2.3}$$

- *Park*:

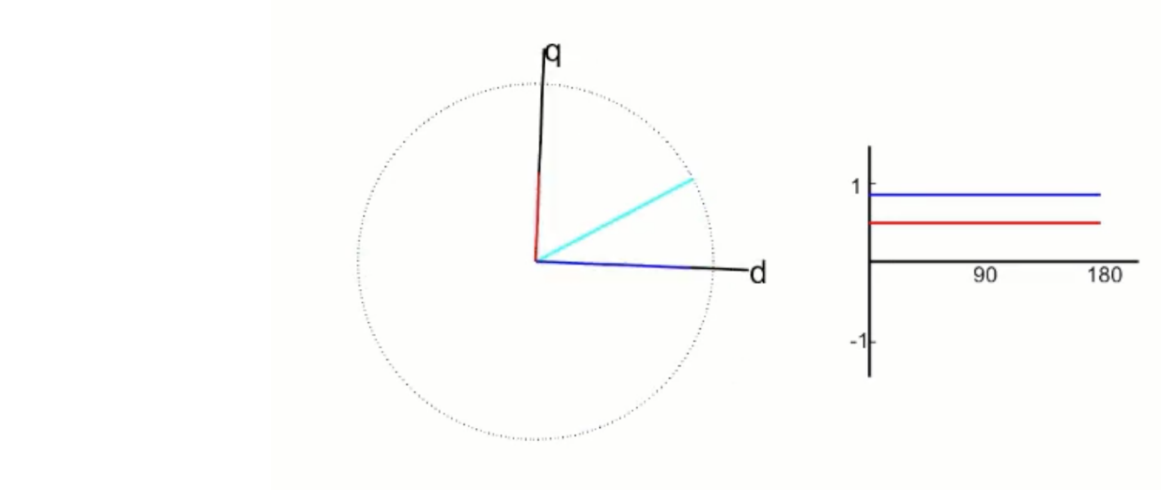


Figure 2.19 - Park transform

The Park transform converts the two components of the αβ frame into an orthogonal rotating reference frame (dq):

$$\begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} \; ; \; \begin{bmatrix} x_d \\ x_q \end{bmatrix} \quad \rightarrow \quad \begin{matrix} x_{\alpha\beta} = x_\alpha + j \cdot x_\beta \\ x_{dq} = x_d + j \cdot x_q \end{matrix} \tag{2.4}$$

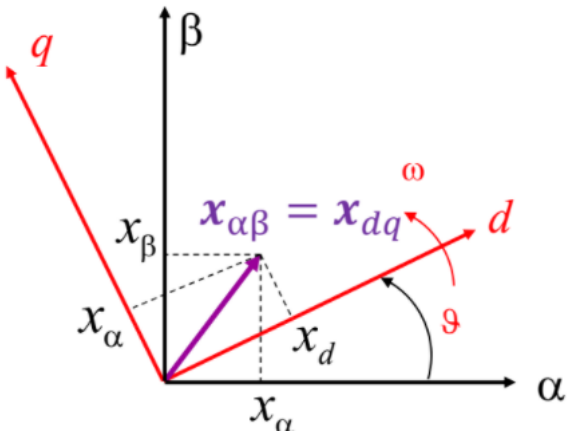


Figure 2.20 - dq reference

x_{dq} is rotated by the angle ϑ respect to $x_{\alpha\beta}$:

$$x_{dq} = e^{-j\theta} x_{\alpha\beta} \qquad x_{\alpha\beta} = e^{j\theta} x_{dq} \tag{2.5}$$

Remembering that $e^{j\theta} = cos\theta + jsin\theta$, the rotation transformation can be written in matrix form:

$$\begin{bmatrix} x_d \\ x_q \end{bmatrix} = \begin{bmatrix} cos\vartheta & sin\vartheta \\ -sin\vartheta & cos\vartheta \end{bmatrix} \cdot \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = [A(\vartheta)] \cdot \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} \tag{2.6}$$

and its inverse:

$$\begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = \begin{bmatrix} cos\vartheta & -sin\vartheta \\ sin\vartheta & cos\vartheta \end{bmatrix} \cdot \begin{bmatrix} x_d \\ x_q \end{bmatrix} = [A(-\vartheta)] \cdot \begin{bmatrix} x_d \\ x_q \end{bmatrix} \tag{2.7}$$

The choice of the angle ϑ influences the definition of the reference system of the transform.

2.5.2 Voltage equations:

The machine model can be written in phase coordinates (abc) using the following variables:

- v_{as}, v_{bs}, v_{cs} – stator phase voltage defined in the stator (abc) frame;
- i_{as}, i_{bs}, i_{cs} – stator phase current defined in the stator (abc) frame;
- $\lambda_{as}, \lambda_{bs}, \lambda_{cs}$ – stator phase flux linkages with the stator windings, defined in the stator (abc) frame;

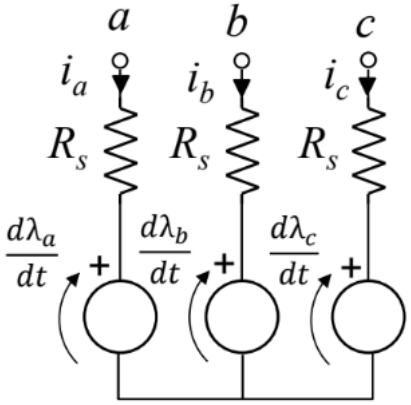


Figure 2.21 - Machine model

The voltage in each phase is given by two contributions: voltage drop on the stator resistance (Joule Losses) and flux linkage variation (induced electro-motive force):

$$\begin{cases} v_a = R_s \cdot i_a + \dfrac{d\lambda_a}{dt} \\ v_b = R_s \cdot i_b + \dfrac{d\lambda_b}{dt} \\ v_c = R_s \cdot i_c + \dfrac{d\lambda_c}{dt} \end{cases}$$

where:

- R_s is the stator phase resistance;
- The subscript "s" is omitted for stator voltage, current and flux for simplicity.

This model can be written in a matrix form:

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = R_s \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \lambda_a \\ \lambda_b \\ \lambda_c \end{bmatrix} \qquad (2.8)$$

The EMF are equivalent to three voltage generators, imposing a voltage equal to the flux linkage derivative (speed dependent).

Then, we can apply the direct Clarke transformation matrix $[T]$ neglecting the homopolar component (not involved in electromechanical energy conversions):

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = R_s \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{d}{dt}\begin{bmatrix} \lambda_a \\ \lambda_b \\ \lambda_c \end{bmatrix} \rightarrow \quad [T]\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = R_s \cdot [T] \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{d}{dt}[T]\begin{bmatrix} \lambda_a \\ \lambda_b \\ \lambda_c \end{bmatrix} \tag{2.9}$$

The voltage equations in $\alpha\beta$ reference frame is therefore obtained:

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = R_s \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \frac{d}{dt}\begin{bmatrix} \lambda_\alpha \\ \lambda_\beta \end{bmatrix} \quad \leftrightarrow \quad v_{\alpha\beta} = R_s \cdot i_{\alpha\beta} + \frac{d}{dt}\lambda_{\alpha\beta} \tag{2.10}$$

we can now rotate this equation to dqreference frame (Park transformation):

$$v_{\alpha\beta} = R_s \cdot i_{\alpha\beta} + \frac{d}{dt}\lambda_{\alpha\beta} \quad \rightarrow \qquad v_{dq} \cdot e^{j\theta} = R_s \cdot i_{dq} \cdot e^{j\theta} + \frac{d}{dt}(\lambda_{dq} \cdot e^{j\theta})$$

then, separately compute the rotation of flux derivative:

$$\frac{d}{dt}\left(\lambda_{dq} e^{j\theta}\right) = \frac{d}{dt}\lambda_{dq} e^{j\theta} + j\frac{d\theta}{dt}\lambda_{dq} e^{j\theta}$$

finally, we obtain:

$$v_{dq} = R_s i_{dq} + \frac{d}{dt}\lambda_{dq} + j\omega\lambda_{dq} \qquad \rightarrow \qquad \begin{cases} v_d = R_s i_d + \frac{d\lambda_d}{dt} - \omega \cdot \lambda_q \\ v_q = R_s i_q + \frac{d\lambda_q}{dt} + \omega \cdot \lambda_d \end{cases} \tag{2.11}$$

in a compact form, we can write:

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = R_s \cdot \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \frac{d}{dt}\begin{bmatrix} \lambda_d \\ \lambda_q \end{bmatrix} + \omega \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} \lambda_d \\ \lambda_q \end{bmatrix} \tag{2.12}$$

The voltage equations are valid for all synchronous machine types (SPM, IPM, SyR and PM-SyR).
The voltage vector in dq reference frame consists of three terms:

$$v_{dq} = R_s i_{dq} + \frac{d}{dt}\lambda_{dq} + j\omega\lambda_{dq} \tag{2.13}$$

where the three terms of the equation are related respectively to the voltage drop on stator resistance (Joule losses), dq flux variation (load variation) and motional term (speed dependent).

2.5.3 Magnetic Model:

The magnetic model of a synchronous machine is essentially related to the rotor. Initially, we will consider ideal iron with infinite magnetic permeance and neglect the magnetic saturation:

- The only reluctance in given by the airgap and the PM ($\mu_{PM} \approx \mu_o$);
- The current-to-flux relationship can be written in terms of (constant) inductances.

The magnetic model in phase coordinates can be written as:

$$\begin{cases} \lambda_a = \lambda_a(i_a, i_b, i_c, \theta, \lambda_m) \\ \lambda_b = \lambda_b(i_a, i_b, i_c, \theta, \lambda_m) \\ \lambda_c = \lambda_c(i_a, i_b, i_c, \theta, \lambda_m) \end{cases}$$

where:

- λ_m is the PM flux linkage (varies with temperature);
- The phase flux linkages depend on all stator current (MMF vector) and on the rotor position θ (PM direction respect to each phase + anisotropy).

The magnetic model of a synchronous machine can be written in phase coordinates as:

$$\begin{bmatrix} \lambda_a \\ \lambda_b \\ \lambda_c \end{bmatrix} = L_{ls} \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} M_{aa} & M_{ab} & M_{ac} \\ M_{ba} & M_{bb} & M_{bc} \\ M_{ca} & M_{cb} & M_{cc} \end{bmatrix} \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \lambda_m \cdot \begin{bmatrix} \cos(\theta) \\ \cos\left(\theta - \frac{2\pi}{3}\right) \\ \cos\left(\theta + \frac{2\pi}{3}\right) \end{bmatrix} \qquad (2.14)$$

where:

- L_{ls} is the leakage phase inductance;
- $M_{ij},\ i = a, b, c,\ j = a, b, c$ are the magnetizing self and mutual inductances.
- $L_{ls} \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix}$ are the Leakage flux linkages;

- $\begin{bmatrix} M_{aa} & M_{ab} & M_{ac} \\ M_{ba} & M_{bb} & M_{bc} \\ M_{ca} & M_{cb} & M_{cc} \end{bmatrix} \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix}$ are the Magnetizing flux linkages;

- $\lambda_m \cdot \begin{bmatrix} \cos(\theta) \\ \cos\left(\theta - \frac{2\pi}{3}\right) \\ \cos\left(\theta + \frac{2\pi}{3}\right) \end{bmatrix}$ are the magnets flux contributions

Due to the rotor anisotropy, the magnetizing inductances are not constant, they depend on the rotor position.
The self-magnetizing inductances are depending on 2α, where α is the angular rotor coordinate.
At first, we can assume the inductance variation along the airgap to be sinusoidal:

$$L(\alpha) = M_{avg} + M_{\Delta} \cdot \cos(2\alpha) \qquad (2.15)$$

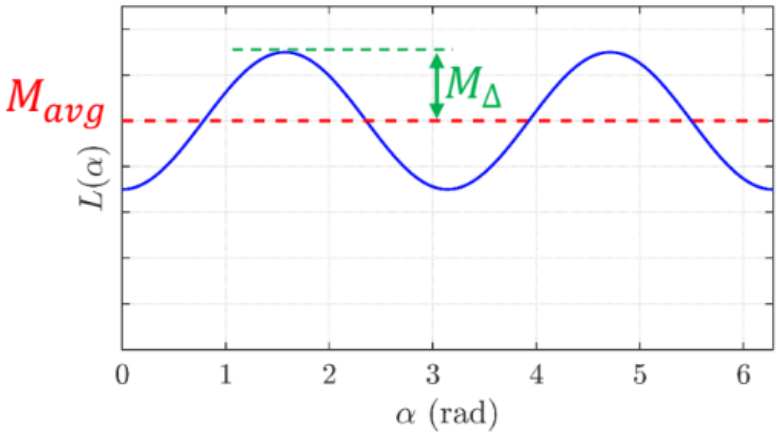


Figure 2.22 - Induction variance

where:

- $M_{avg} = \frac{M_d + M_q}{2}$ is the average inductance;

- $M_{\Delta} = \frac{M_d - M_q}{2}$ is the differential inductance due t the rotor anisotropy ($M_{\Delta} < 0$).

The three-phase self-inductance are obtained by considering the angle displacement of the abc axes respect to α:

- Phase a: $\alpha = -\theta$
- Phase b: $\alpha = \frac{2\pi}{3} - \theta$
- Phase c: $\alpha = \frac{4\pi}{3} - \theta$

then, we obtain:

$$M_{aa} = M_{avg} + M_{\Delta} \cdot \cos(2\theta)$$
$$M_{bb} = M_{avg} + M_{\Delta} \cdot \cos\left(2\theta + \frac{2\pi}{3}\right)$$
$$M_{cc} = M_{avg} + M_{\Delta} \cdot \cos\left(2\theta - \frac{2\pi}{3}\right)$$

similarly, the mutual inductances also depend on 2θ:

$$M_{ab} = M_{ba} = -\frac{1}{2}M_{avg} + M_{\Delta} \cdot \cos\left(2\theta - \frac{2\pi}{3}\right)$$
$$M_{ac} = M_{ca} = -\frac{1}{2}M_{avg} + M_{\Delta} \cdot \cos\left(2\theta + \frac{2\pi}{3}\right)$$
$$M_{bc} = M_{cb} = -\frac{1}{2}M_{avg} + M_{\Delta} \cdot \cos(2\theta)$$

therefore, each term of the magnetizing inductance matrix depends on 2θ:

$$\begin{bmatrix} M_{aa} & M_{ab} & M_{ac} \\ M_{ba} & M_{bb} & M_{bc} \\ M_{ca} & M_{cb} & M_{cc} \end{bmatrix} = [M(2\theta)] \qquad (2.16)$$

The magnetizing inductance matrix in 3-phase coordinates is composed by two terms (commode mode + differential mode):

$$[M(2\theta)] = \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{2} & 1 \end{bmatrix} \cdot M_{avg} + \begin{bmatrix} \cos(2\theta) & \cos\left(2\theta - \frac{2\pi}{3}\right) & \cos\left(2\theta + \frac{2\pi}{3}\right) \\ \cos\left(2\theta - \frac{2\pi}{3}\right) & \cos\left(2\theta + \frac{2\pi}{3}\right) & \cos(2\theta) \\ \cos\left(2\theta + \frac{2\pi}{3}\right) & \cos(2\theta) & \cos\left(2\theta - \frac{2\pi}{3}\right) \end{bmatrix} \cdot M_{\Delta} \quad (2.17)$$

The complete magnetic model (inductance matrix + PM flux) in phase coordinate is:

$$\begin{bmatrix}\lambda_a\\ \lambda_b\\ \lambda_c\end{bmatrix}=\begin{bmatrix} L_{ls}+M_{avg}+M_\Delta\cdot\cos(2\theta) & -\frac{1}{2}M_{avg}+M_\Delta\cdot\cos\left(2\theta-\frac{2\pi}{3}\right) & -\frac{1}{2}M_{avg}+M_\Delta\cdot\cos\left(2\theta+\frac{2\pi}{3}\right)\\ -\frac{1}{2}M_{avg}+M_\Delta\cdot\cos\left(2\theta-\frac{2\pi}{3}\right) & L_{ls}+M_{avg}+M_\Delta\cdot\cos\left(2\theta+\frac{2\pi}{3}\right) & -\frac{1}{2}M_{avg}+M_\Delta\cdot\cos(2\theta)\\ -\frac{1}{2}M_{avg}+M_\Delta\cdot\cos\left(2\theta+\frac{2\pi}{3}\right) & -\frac{1}{2}M_{avg}+M_\Delta\cdot\cos(2\theta) & L_{ls}+M_{avg}+M_\Delta\cdot\cos\left(2\theta-\frac{2\pi}{3}\right)\end{bmatrix}\cdot\begin{bmatrix}i_a\\ i_b\\ i_c\end{bmatrix}+\lambda_m\cdot\begin{bmatrix}\cos(\theta)\\ \cos\left(\theta-\frac{2\pi}{3}\right)\\ \cos\left(\theta+\frac{2\pi}{3}\right)\end{bmatrix} \quad (2.18)$$

The magnetic model can be conveniently transformed in $(\alpha,\beta,0)$ frame by applying the Clark transformation matrix:

$$\begin{bmatrix}\lambda_a\\ \lambda_b\\ \lambda_c\end{bmatrix}=L_{ls}\cdot\begin{bmatrix}i_a\\ i_b\\ i_c\end{bmatrix}+[M(2\theta)]\begin{bmatrix}i_a\\ i_b\\ i_c\end{bmatrix}+\lambda_m\cdot\begin{bmatrix}\cos(\theta)\\ \cos\left(\theta-\frac{2\pi}{3}\right)\\ \cos\left(\theta+\frac{2\pi}{3}\right)\end{bmatrix} \quad (2.19)$$

multiplying both side by [T] :

$$[T]\begin{bmatrix}\lambda_a\\ \lambda_b\\ \lambda_c\end{bmatrix}=L_{ls}[T]\begin{bmatrix}i_a\\ i_b\\ i_c\end{bmatrix}+[T][M(2\theta)]\begin{bmatrix}i_a\\ i_b\\ i_c\end{bmatrix}+\lambda_m[T]\begin{bmatrix}\cos(\theta)\\ \cos\left(\theta-\frac{2\pi}{3}\right)\\ \cos\left(\theta+\frac{2\pi}{3}\right)\end{bmatrix}$$

we obtain that:

$$\begin{bmatrix}\lambda_\alpha\\ \lambda_\beta\\ \lambda_o\end{bmatrix}=L_{ls}\begin{bmatrix}i_\alpha\\ i_\beta\\ i_o\end{bmatrix}+[T][M(2\theta)][T]^{-1}\begin{bmatrix}i_\alpha\\ i_\beta\\ i_o\end{bmatrix}+\lambda_m[T]\begin{bmatrix}\cos(\theta)\\ \cos\left(\theta-\frac{2\pi}{3}\right)\\ \cos\left(\theta+\frac{2\pi}{3}\right)\end{bmatrix}$$

the matrix products can be computed separately:

$$[T][M(2\theta)][T]^{-1}=\begin{bmatrix}\frac{3}{2}M_{avg}+\frac{3}{2}M_\Delta\cdot\cos(2\theta) & \frac{3}{2}M_\Delta\cdot\sin(2\theta) & 0\\ \frac{3}{2}M_\Delta\cdot\sin(2\theta) & \frac{3}{2}M_{avg}-\frac{3}{2}M_\Delta\cdot\cos(2\theta) & 0\\ 0 & 0 & 0\end{bmatrix}$$

$$[T]\begin{bmatrix}\cos(\theta)\\ \cos\left(\theta-\frac{2\pi}{3}\right)\\ \cos\left(\theta+\frac{2\pi}{3}\right)\end{bmatrix}=\begin{bmatrix}\cos(\theta)\\ \sin(\theta)\\ 0\end{bmatrix}$$

we keep only the (α,β) components:

$$\begin{bmatrix}\lambda_\alpha\\ \lambda_\beta\end{bmatrix}=\left\{\left(L_{ls}+\frac{3}{2}M_{avg}\right)\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}+\frac{3}{2}M_\Delta\begin{bmatrix}\cos(2\theta) & \sin(2\theta)\\ \sin(2\theta) & -\cos(2\theta)\end{bmatrix}\right\}\cdot\begin{bmatrix}i_\alpha\\ i_\beta\end{bmatrix}+\lambda_m\cdot\begin{bmatrix}\cos(\theta)\\ \sin(\theta)\end{bmatrix}$$

The magnetic model referred to the rotor (d,q) frame is obtained using the rotation transformation:

$$[A(\theta)]\begin{bmatrix}\lambda_\alpha\\ \lambda_\beta\end{bmatrix}=[A(\theta)]\left\{\left(L_{ls}+\frac{3}{2}M_{avg}\right)\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}+\frac{3}{2}M_\Delta\begin{bmatrix}\cos(2\theta) & \sin(2\theta)\\ \sin(2\theta) & -\cos(2\theta)\end{bmatrix}\right\}\begin{bmatrix}i_\alpha\\ i_\beta\end{bmatrix}+\lambda_m[A(\theta)]\begin{bmatrix}\cos(\theta)\\ \sin(\theta)\end{bmatrix}$$

we can separately compute the matrix products:

$$[A(\theta)]\left\{\left(L_{ls}+\frac{3}{2}M_{avg}\right)\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}+\frac{3}{2}M_\Delta\begin{bmatrix}\cos(2\theta) & \sin(2\theta)\\ \sin(2\theta) & -\cos(2\theta)\end{bmatrix}\right\}[A(-\theta)]=$$

$$=\begin{bmatrix}L_{ls}+\frac{3}{2}M_{avg}+\frac{3}{2}M_\Delta & 0\\ 0 & L_{ls}+\frac{3}{2}M_{avg}-\frac{3}{2}M_\Delta\end{bmatrix}$$

with:

$$[A(\theta)]\begin{bmatrix}\cos(\theta)\\ \sin(\theta)\end{bmatrix}=\begin{bmatrix}1\\ 0\end{bmatrix}$$

The magnetic model becomes independent of rotor position if referred to the rotor (d,q) frame:

$$\begin{bmatrix}\lambda_d\\ \lambda_q\end{bmatrix}=\begin{bmatrix}L_{ls}+\frac{3}{2}M_{avg}+\frac{3}{2}M_\Delta & 0\\ 0 & L_{ls}+\frac{3}{2}M_{avg}-\frac{3}{2}M_\Delta\end{bmatrix}\begin{bmatrix}i_d\\ i_q\end{bmatrix}+\lambda_m\cdot\begin{bmatrix}1\\ 0\end{bmatrix}\qquad(2.20)$$

where:

- $L_d=L_{ls}+\frac{3}{2}M_{avg}+\frac{3}{2}M_\Delta$ → magnetizing inductance in d axis (min L);

- $L_q = L_{ls} + \frac{3}{2}M_{avg} - \frac{3}{2}M_{\Delta}$ → magnetizing inductance in q axis (max L);

Final magnetic **PMSM motor model** in (d, q) frame :

$$\begin{bmatrix}\lambda_d \\ \lambda_q\end{bmatrix} = \begin{bmatrix}L_s & 0 \\ 0 & L_s\end{bmatrix} \cdot \begin{bmatrix}i_d \\ i_q\end{bmatrix} + \begin{bmatrix}\lambda_m \\ 0\end{bmatrix} \qquad (2.21)$$

The machine is theoretically isotropic, so $L_d = L_q = L_s$, where L_s is called synchronous inductance.

2.5.4: PMSM Operation regions

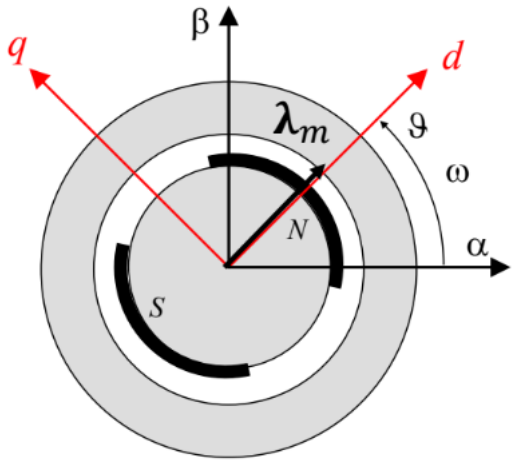


Figure 2.23 - Operation region of PMSM

In this last part of the chapter, we will make an overview of the operating regions of a permanent magnet synchronous motor (PMSM), in order to understand how the engine is piloted and controlled. In particular, an isotropic machine will be addressed, which is the typology of the motor used in this thesis.
The control of a permanent magnet synchronous motor (PMSM) is closely linked to the technical characteristics of the engine itself (current, speed, nominal values of voltage) and of the inverter that is used to drive it.
All these parameters define what are the limits that must be satisfied at the same time. Furthermore, even high-speed operation can introduce restrictions as the counter-electromotive forces that occur in the stator windings are proportional to the speed, and therefore they can exceed the sustainable values of both the machine and the power stage. The control strategies of the PMSM therefore adapt to these restrictions and from time to time adapt their behavior to the particular working condition.
Initially, we will consider the general case of PMSM, later we will analyze in detail the case of an isotropic engine $(L_d = L_q)$.
If we return to the dynamic equations of the PMSM model previously explained, we can obtain the following steady-state equations:

$$\begin{cases} v_d = R_s i_d - \omega L_q I_q \\ v_d = R_s i_q + \omega L_d i_d + \omega \lambda_m \end{cases} \qquad (2.22)$$

$$T = \frac{3}{2} p\left[\lambda_m i_q + (L_d - L_q) i_d i_q\right] \qquad (2.23)$$

If we may assume that the winding resistances are very small so as to be able to neglect the relative voltage drops, then the equations (2.22) & (2.23) become:

$$\begin{cases} v_d = -\omega L_q I_q \\ v_d = \omega L_d i_d + \omega \lambda_m \end{cases} \qquad (2.24)$$

$$T = \frac{3}{2} p\left[\lambda_m i_q + (L_d - L_q) i_d i_q\right] \qquad (2.25)$$

In these last equations, both the voltages and the currents must assume values that fall within the range of nominal values. These restrictions identify very specific regions of operation; since all the limits must be satisfied at the same time, the set of admissible values for the PMSM during steady-state operation is obtained from the intersection of these regions.

As regards the current limit, it is identified by the maximum modulus (of the space vector) of the current (nominal value I_N) which can be suppressed in steady state:

$$I = \sqrt{I_d^2 + I_q^2} \leq I_N \qquad (2.26)$$

As shown in the figure below, this latter condition identifies in the current plane $I_{d,} I_q$ a circular region (current limit), of radius equal to I_N, centered at the origin.

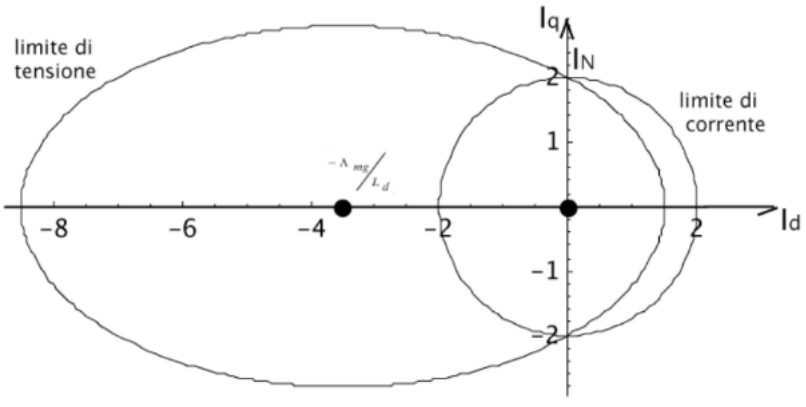


Figure 2.24 - Circle limit of operation

Similarly, the voltage space vector module cannot exceed the nominal value, that depends on the maximum voltage at which the drive can operate due to the limits of the inverter or motor:

$$V = \sqrt{v_d^2 + v_q^2} \leq V_N \qquad (2.27)$$

Substituting the steady-state equations of vd, vq in the last equation(2.24), we obtain:

$$\left(I_d + \frac{\lambda_m}{L_d}\right)^2 + \left(\frac{L_d}{L_q}\right)^2 I_q^2 \leq \frac{V_N^2}{\omega_{em}^2 L_d^2} \qquad (2.28)$$

The latter equation, depending on the speed of traction ω, describes in the plan of currents $I_{d,}I_q$ a family of ellipses having center C, major semi-axis r and eccentricity ξ (also called salience factor) respectively of:

$$C = \left(-\frac{\lambda_m}{L_d}, 0\right) \ (2.29)$$

$$r = \frac{V_N}{\omega_m L_d} \qquad (2.30)$$

$$\xi = \frac{L_q}{L_d} \qquad (2.31)$$

Voltage drops across phase resistors only affect voltage limits. These limits restrict the region of operation at high speeds where, however, the inductive part of the motor impedance prevails.
This consideration allows us to further justify the simplification made previously.
In the case of the isotropic machine the inductances relative to the d axis and the q axis are equal ($L_d = L_q = L, \xi = 1$)and therefore the equation becomes:

$$\begin{cases} v_d = -\omega L I_q \\ v_d = \omega L i_d + \omega \lambda_m \end{cases} \qquad (2.32)$$

$$T = \frac{3}{2} p\left[\lambda_m i_q\right] \qquad (2.33)$$

The equation it describes a family of ellipses, previously mentioned, therefore represents a family of circles with center in $\left(-\frac{\lambda_m}{L}, 0\right)$ and radius proportional to the ratio between the applied voltage and the rotation speed; the center of the circumferences can be both inside and outside the current limit curve.
The constant torque curves are, in the plane $I_{d,}I_q$, lines parallel to the d axis:

$$I_q = \frac{T}{\frac{3}{2}p\lambda_m} \qquad (2.34)$$

Since the torque depends only on $I_q$it is convenient to work with $I_d = 0$ thus obtaining maximum torque with the same current module. In this way the control system (field-oriented) operates in the region known as constant available torque, characterized by the equation:

$$\begin{cases} v_d = -\omega L I_q \\ v_d = \omega\lambda_m \end{cases} \qquad (2.35)$$

$$T = \frac{3}{2}p\left[\lambda_m i_q\right] \qquad (2.36)$$

The two figures shown below, represent the constant available torque working point, respectively for $\left(\frac{\lambda_m}{L} > I_N\right)$ and $\left(\frac{\lambda_m}{L} < I_N\right)$:

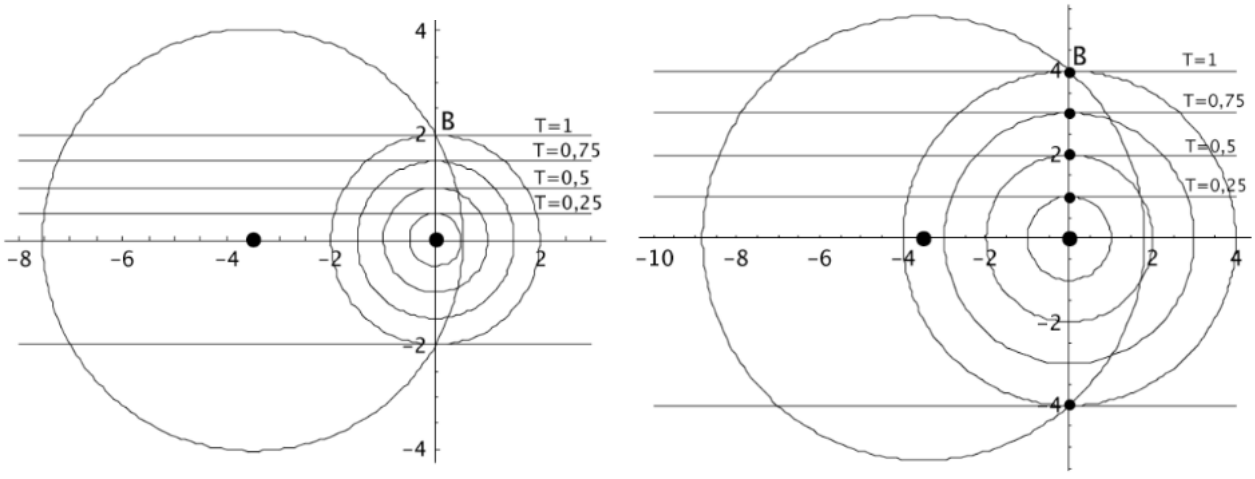


Figure 2.25 - Circular diagram for an isotropic PMSM (SPMa, SPMb)

With the same torque, the speed is proportional to the module of the applied voltage and reaches the nominal value, called base speed of the drive, corresponding to the nominal voltage; with an appropriate motor design it is possible to make the nominal voltage (at nominal speed) and current limit curves intersect on the $\mathrm{I_q}$ axis (point B in the figure above, corresponding to the basic quantities in the normalized model) thus ensuring the rated torque up to the rated speed.

To see what the base speed ω_b is, just consider the triangle AOB in the following figure:

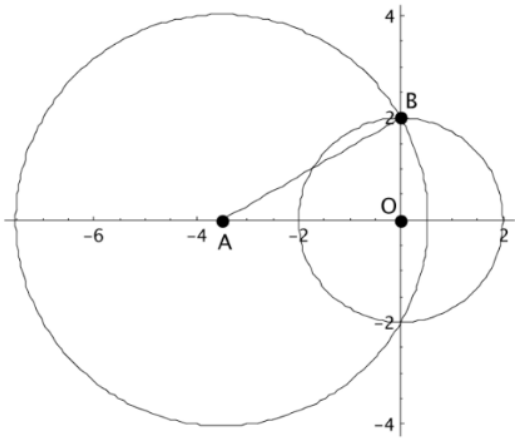


Figure 2.26 - Triangle AOB for calculate ω_b

we get that:

$$\frac{V_N^2}{\omega_b^2 L^2} = I_N^2 + \left(\frac{\lambda_m}{L}\right)^2 \qquad (2.37)$$

from which it is derived:

$$\omega_b = \frac{V_N}{\sqrt{L^2 I_N^2 + \lambda_m^2}} \qquad (2.38)$$

We therefore distinguish the following two cases:

- $\omega_m < \omega_b$: the radius will be larger than the one that generates the tension limit circumference and therefore outside the limit. In this condition, the motor delivers maximum torque which distinguishes the constant torque zone.
  This region is therefore characterized by the zero-current I_d, from I_q between $-I_N$and I_N, and by the variable voltage proportionally to the speed up to the value V_N, which is reached by the rated speed $\omega_{m,N}$. At zero speed the current is called the state current, like the torque called the stall torque.

- $\omega_m > \omega_b$: the radius is lower than the limit, so it is no longer possible to work in the constant torque area and we are forced to reduce the maximum possible torque, we work in the constant apparent power area.

To increase the motor speed beyond the nominal value, since it is not possible to increase the voltage (limit value), we operate in *"flux-weakening".*
As the speed ω_m increases, the amplitude of the limited voltage circumferences decreases and therefore the optimal working point (maximum torque with respect to the current used) moves in the $I_{d,}I_q$ plane, corresponding to $I_{d,}$values that are no longer null. Consequently, a flux component is generated which opposes to that produced by the permanent magnet. In this way, similar result is obtained when the excitation current is decreased.
Two cases have to be considered mainly:

1. *The center of the voltage limit circles is outside the current limit*:

In this situation$\left(\frac{\lambda_m}{L} = I_N\right)$, the point that describes the highest torque / current ratio (with the same voltage) remains on the current limit circumference and lies on the intersection with the voltage limit circumference corresponding to the rotation speed (constant apparent power region). This speed can reach the maximum theoretical value:

$$\frac{V_N}{\omega_{m,max}L} + I_N = \frac{\lambda_m}{L} \quad \rightarrow \quad \omega_{m,max} = \frac{V_N}{\lambda_m - LI_N} \qquad (2.39)$$

in correspondence of which the current limit circle is tangent to the voltage limit circle at a point on the I_d axis (and which corresponds to zero available torque and power)

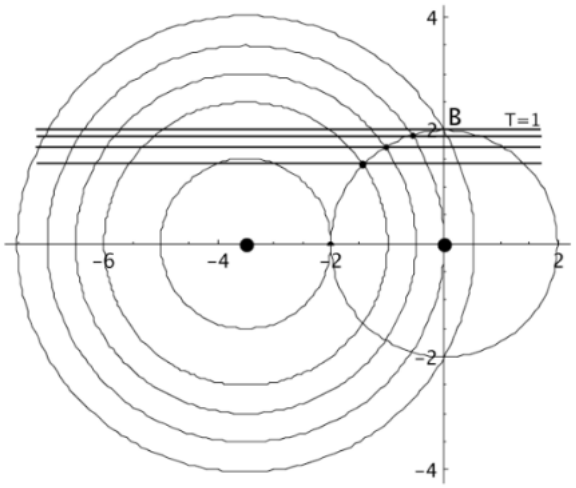


Figure 2.27 - Work points in flux-weakening (SPMa)

The trends of the torque, voltage and currents (I_d, I_q and module I) for an isotropic machine with the center of the voltage limit circles outside the current limit circle are represented by the following figures:

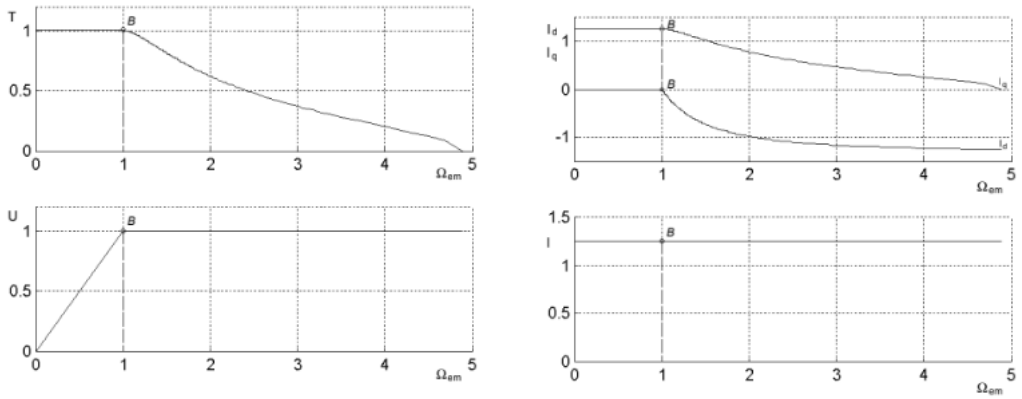


Figure 2.28 – Isotropic PMSM (SPMa), Torque e voltage (left), Current Idq and I modulus (right)

Once the base speed has been reached, the voltage remains constant and equal to the nominal value while the torque decreases reaching the zero value when the speed assumes the maximum value; the modulus of the currents I remains constant as I_q decreases (which guarantees the limit) and as $I_d$increases in absolute value (to deflux the machine).

2. *<u>The center of the voltage limit circles is inside the current limit</u>*

In this situation $\left(\frac{\lambda_m}{L} < I_N\right)$, the point that describes the greatest torque / current ratio (with the same voltage) initially remains on the current limit circumference, as in the previous case (region with constant apparent power). Once the point P is reached (on the vertical of the center of the voltage limit circles), it follows the straight-line having abscissa $-\frac{\lambda_m}{L}$ (region with decreasing apparent power). Moreover, in this situation there is no speed limit of rotation with the exception due to the restrictions imposed by the mechanics of the machine.

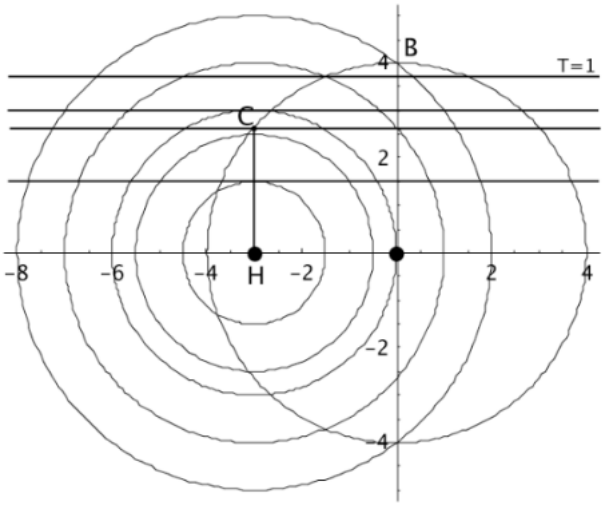


Figure 2.29 - Work points in flux-weakening (SPMb)

The variable I control when switching from one system to another (i.e. from BC to CH) is the speed. Depending on the speed I decide what value to assign to I_d^*:

- $\omega \leq \omega_b$: I assign the necessary value to I_q^* and place $I_d^* = 0$;
- $\omega_b < \omega \leq \omega_{CH}$: I_d^* = circumference;
- $\omega > \omega_{CH}$: $I_d^* = -\frac{\lambda_m}{L}$ constant value.

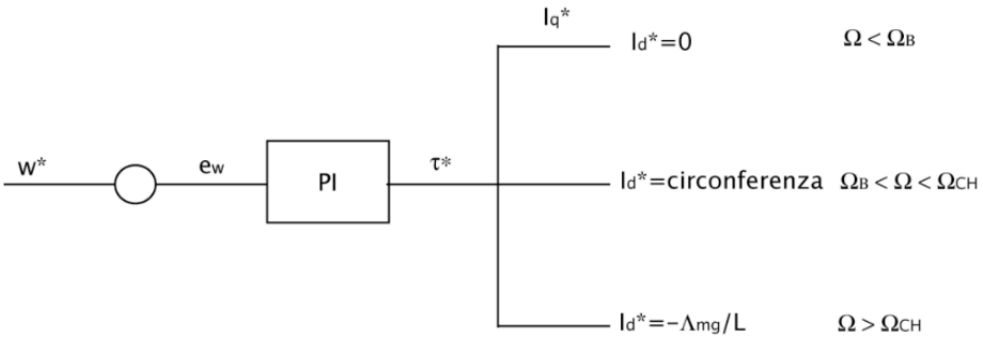


Figure 2.30 - Block control scheme

The control of $I_q^*$is feedback, while for id it is necessary to calculate the speed. To calculate ω_{CH} proceed as follows:

$$I_N^2 = \left(\frac{\lambda_m}{L}\right)^2 + \left(\frac{V_N}{\omega_{CH} L}\right)^2 \rightarrow \omega_{CH} = \sqrt{\frac{V_N^2}{(LI_N)^2 - \lambda_m^2}} \qquad (2.40)$$

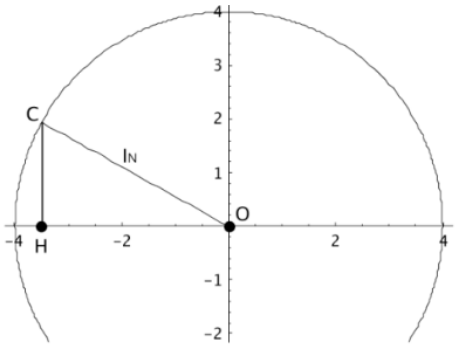


Figure 2.31 - Triangle CHO for calculateω_{ch}

It is important to understand the proper flux-weakening technique. You have to act on the speed to then have effects on the torque, the cause-effect principle (torque-speed) also helps in this case: you have to intervene on the effects to modify the causes.

The trends of the torque, voltage and currents (I_d, I_q and module I) for an isotropic machine with the center of the voltage limit circles inside the current limit circle are represented by the following figures:

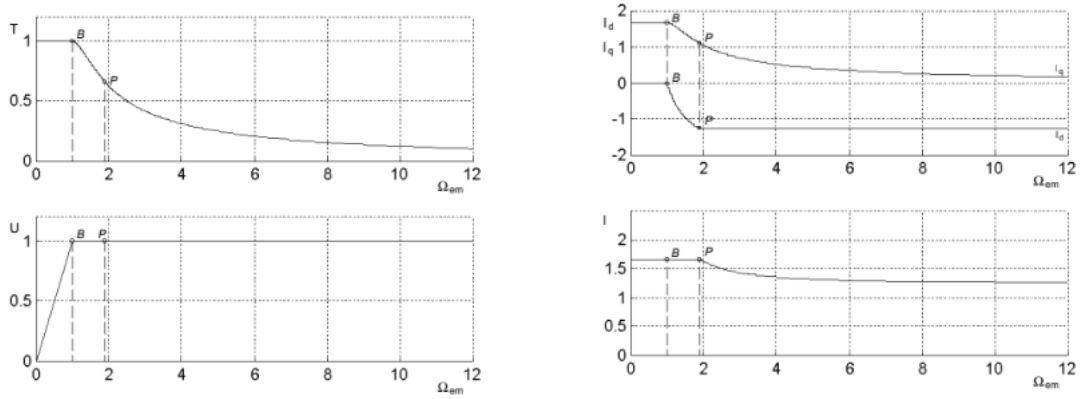


Figure 2.32 – Isotropic PMSM (SPMb), Torque e voltage (left), Current Idq and I modulus (right)

Reached the base speed, the voltage remains constant and equal to the nominal value while the torque decreases keeping asymptotically at zero. The current modulus I initially remains constant (from point B to P) and then decreases in the region with decreasing apparent power up to the value $\frac{\lambda_m}{L}$; the current component I_d increases from B to P (to weaken the machine) and then remains constant while the current component I_q tends asymptotically to zero.

2.6 Controller

Nowadays, electrical motors and other rotating machines have become dependent on the presence of electrical drives. We can distinguish three different types of work that electrical drives can perform:

1. Starting;
2. Speed Control;
3. Braking;

Control of electrical drive is an important task because all the functions accomplished by the drives are principally transient operations. The change in terminal voltage or in current are huge, which may damage the motor temporarily or permanently.
In general, Electrical drives enable us to control the motor in every aspect.

2.6.1 Closed loop Control of Drives

Closed-loop system is a feedback system, it has the ability to modify and particularly to stabilize the natural dynamics of a system. We will now analyze the advantages (and disadvantages) of closed-loop feedback control over any other type of control architecture. The purpose of any electrical or electronic control system is to measure, monitor, and control a process. A way we can control the process is by monitoring its output and "feeding" some of it back, in order to compare the desired output with the actual output to reduce the risk of error and even if disturbed, to bring the output of the system back to the original or desired response.
One of the main characteristics of a Closed-loop Control System, or feedback control system, is that it uses the concept of an open loop system as its forward path but has one or more feedback loops (hence its name) or paths between its output and its input. The term "feedback" means that some portion of the output is turned to the input to form part of the systems excitation and to reduce the error between the actual and the desired signal.

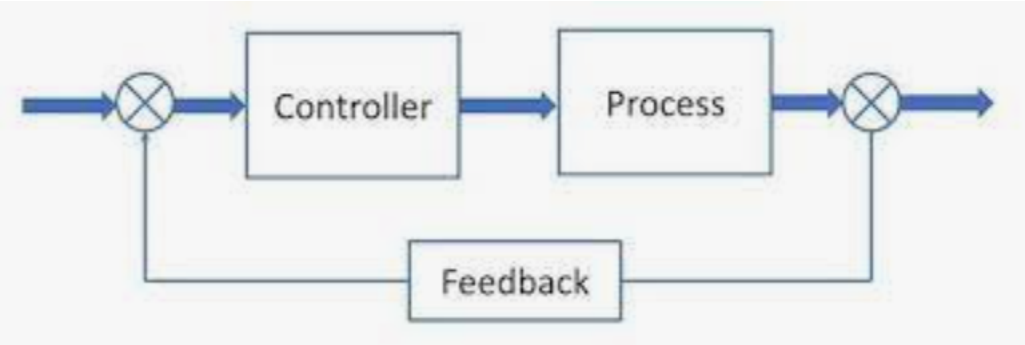


Figure 2.33 - Closed loop control

The main difference between an open-loop and a closed-loop system is that the latter uses a feedback control action in order to reduce any errors within the system.
Output accuracy depends on the feedback path, can be very accurate and within electronic control systems and circuits, feedback control is more commonly used than open-loop or feed forward control.
Closed-loop systems have more advantages respect to the open-loop systems. First of all, a closed-loop feedback control system has the ability to reduce a system's sensitivity to external disturbances.
Then the main characteristics of Closed-loop Control are:

- reduce errors by automatically adjusting the systems input.
- improve stability of an unstable system.
- increase or reduce the system sensitivity.
- enhance robustness against external disturbances to the process.
- produce a reliable and repeatable performance.

Although a good closed-loop system has many more advantages than an open-loop control system, one of its main disadvantages is that the structure, in order to achieve the control of the system, is even more complex by having one or more feedback paths. Furthermore, the system can become unstable and start to oscillate, as the controller tries to over-correct itself, if the gain of the controller is too sensitive to changes in its input commands or signals. In this way, we must define some pre-defined limits in order to make the system behave as we want.
From the controller point of view, the process + Feedback are seen as a plant to be controlled.

Typically, the study of the system is performed in the Laplace domain, using the transfer function obtained from the dynamic models:

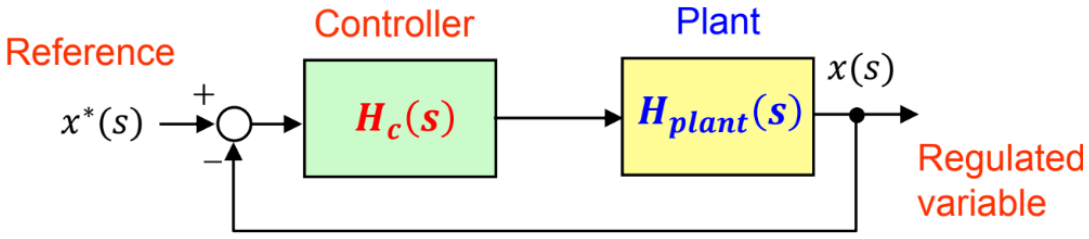


Figure 2.34 - Closed loop control in Laplace domain

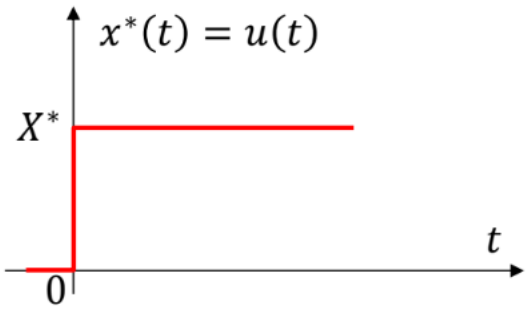
- The known system part (plant) is represented by the transfer function $H_{plant}(s)$ that includes the power converter, its load and the feedback;
- The $H_c(s)$ is the transfer function of the controller to be designed;
- Unity feedback.

Besides the stability, the objectives of the closed-loop control obtained through a proper controller design are:

- Zero steady-state error ($x^*(t) = x(t)$);
- Good dynamic performance with minimal settling times and overshoots;
- Good rejection of disturbances.

Typically, the performance evaluation of a closed-loop system is obtained with two types of variations applied to the input reference:

1. *Step variation* of the reference:



$$u(t) = \begin{cases} 0 \ \ if \ t < 0 \\ X^* \ if \ t \geq 0 \end{cases}$$

Figure 2.35 – Step reference

The performance of a closed-loop system subjected to step variation of the reference is given by:

- *Rising time*: time need for the feedback to reach 95% of the reference after its step variation

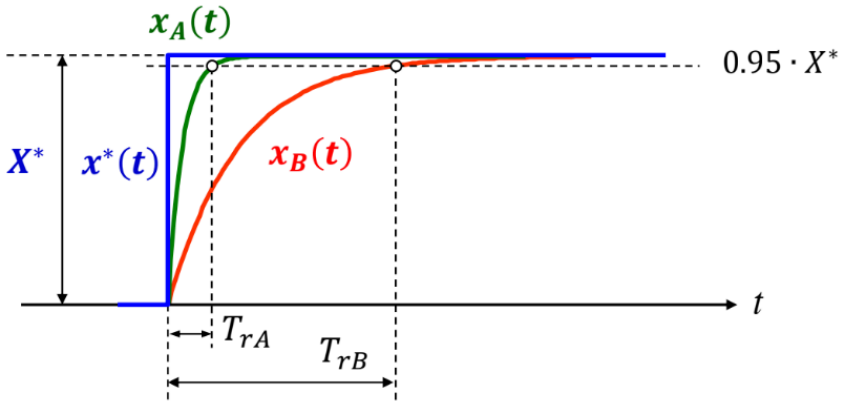


Figure 2.36 – Rising time

- *Settling time*: time after which the output changes less than 5% with respect to the input

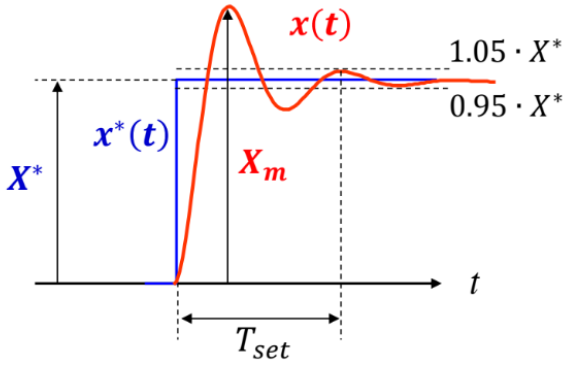


Figure 2.37 – Settling time

- *Percentage overshoot*: maximum deviation of the output with respect to the input

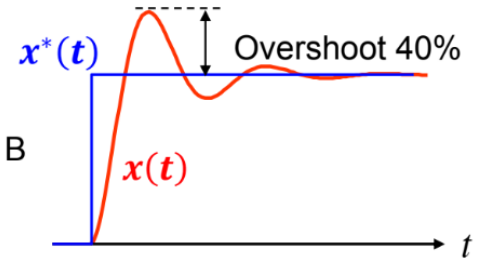


Figure 2.38 – Percentage overshoot

2. Sinusoidal variation of the reference

The principal metric related to a sinusoidal input is called *'system bandwidth* $f_b\ (Hz)$*' for* which the amplitude of the output reduces at 70.7% with respect to the amplitude of the input

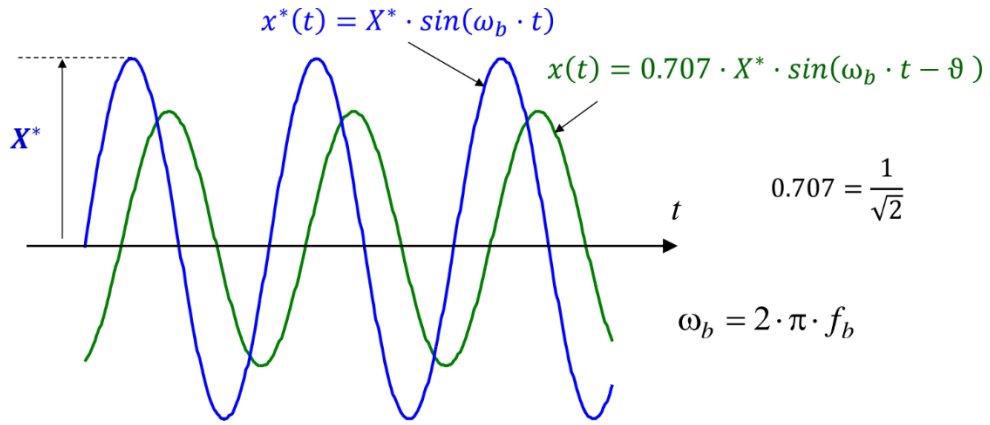


Figure 2.39 – Sinusoidal reference

In general, the system bandwidth is a very good metric to understand the system performance. If low bandwidth is referred to a slow system, a high bandwidth is referred to a fast system with low rising and settling times.
If the bandwidth is too high, the system may become too sensitive to disturbances and the output may oscillate due to the noise in the measurement.
The design of a closed-loop system in power conversion requires an accurate analysis of the system.
Typically, the transfer function of the plant in power conversion, requires systems relatively simple:

$$H_{CL}(s) = \frac{w_o^2}{s^2 + 2 * \xi * w_o * s + w_o^2} = \frac{p_1 * p_2}{(s - p_1) * (s - p_2)}$$

where:

- $w_o = \sqrt{p_1 * p_2}$ is the natural frequency/pulsation ($\frac{rad}{s}$);
- ξ is the damping;
- $p_{1,2}$ are complex conjugate poles having negative real values;

The Closed-loop transfer function is therefore:

$$H_{CL}(s) = \frac{H_{OL}(s)}{1 + H_{OL}(s)} \qquad H_{OL}(s) = H_c(s) \cdot H_{plant}(s)$$

2.6.2 Open loop Control of Drives

Initially, electrical control systems were basically manual or what is called an Open-loop System with very few automatic control or feedback features built in to regulate the process variable so as to maintain the desired output level or value.

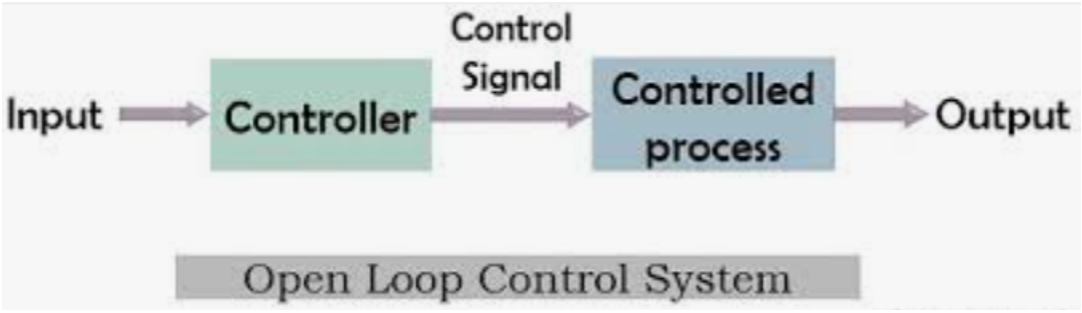


Figure 2.40 - Open loop control

A non-feedback system, also referred to as Open-loop system, is a particular continuous control system where the control action of the input signal does not affect the output of system. Unlike a closed-loop system, in a non-feedback system the output is neither measured nor "fed back" for comparison with the input.
Therefore, regardless of the final result, an open-loop system is expected to follow carefully its input set point or command.
Moreover, this typology of control system has no information of the output condition, so when the preset value drift, it cannot self-correct any errors it could make, even if this results in large deviations from the preset value.
Another handicap of this type of systems is that they are not able to handle disturbances or changes in their condition in a significant way, which may reduce the reliability of the system to complete the desired task.
Then we can resume the main characteristics of an "Non-feedback system" as follows:

- No comparison between actual and desired values.
- No self-regulation or control action over the output value.
- A fixed operating position for the controller is determined by each input setting.
- Changes or disturbances in external conditions does not result in a direct output change (unless the controller setting is altered manually).

Referring to Figure (2.42), the Transfer Function of an open-loop system is the product of each single transfer function of the cascaded blocks in series:

$$H_{OL}(s) = H_c(s) * H_{plant}(s)$$

In an Open-loop system we can distinguish two parameters:

1. Crossover frequency $f_c(Hz)$: is the frequency at which the modulus of the open-loop transfer function is equal to 1 (0 dB)

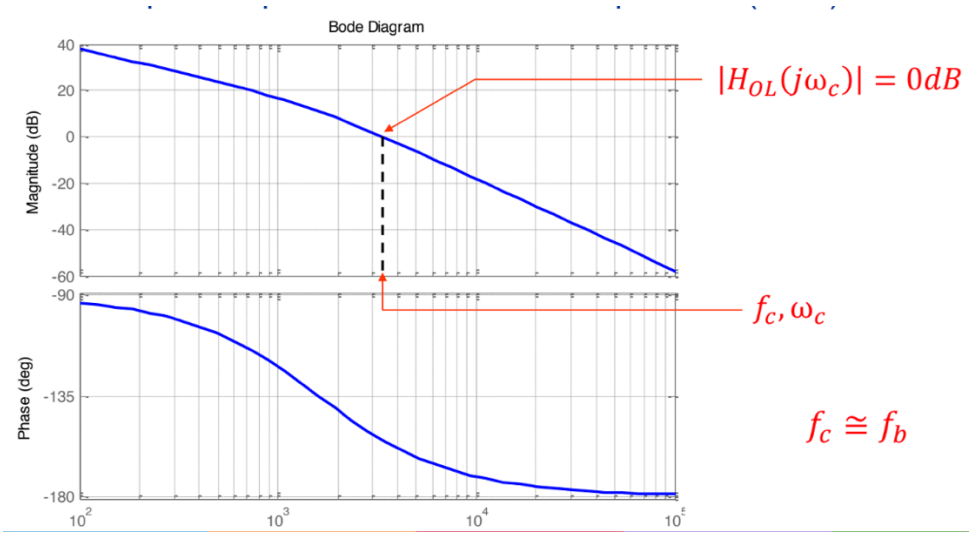


Figure 2.41 – Crossover frequency

At crossover frequency, the phase delay must be lower than 180 electrical degrees for a stable closed-loop system.

2. Phase margin (PM) Φ_{PM}: defined at crossover frequency is

$$\Phi_{PM} = \angle H_{OL}(jw_c) - (-180) = |\Phi_{OL}| + 180(\text{deg})$$

In order to get a good dynamic response without any stability issue, the phase margin must be higher than 45 electrical degrees, a good hint is to get 60 electrical degrees.

2.6.3 Controllers in power electronics and drives

In this chapter, we are going to make a general overview of the characteristic of the most employed controllers.
In particular, we can distinguish:

1. Proportional controller (P):

$$H_c(s) = k_p$$

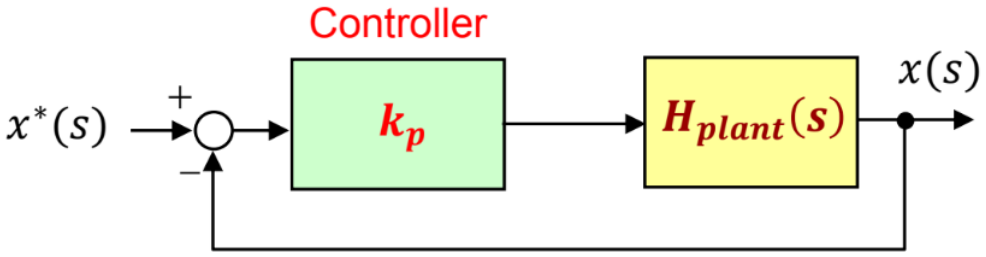


Figure 2.42 – P Controller

2. Proportional-integral controller (PI):

$$H_c(s) = k_p + \frac{k_i}{s}$$

where:

k_p = proportional gain, k_i = integral gain

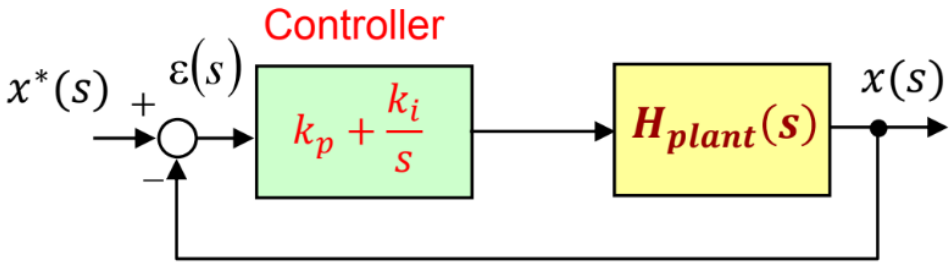


Figure 2.43 – PI Controller

The PI controller is the most employed solution in e-Drives, where typically we have plants with one dominant time constant.

3. Proportional-Integral derivative controller (PID):

$$H_c(s) = k_p + \frac{k_i}{s} + k_d * s$$

Where:

k_p = proportional gain, k_i = integral gain, k_d = derivative gain

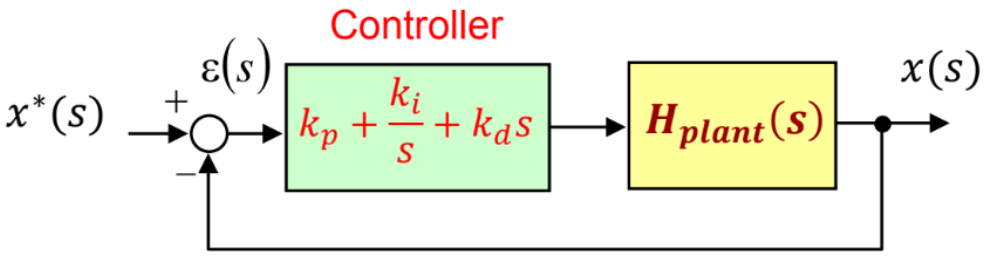


Figure 2.44 – PID Controller

Typically, the PID controller can be used for a plant with two dominant time constants. The PID controller must be used with caution: the noise in the current acquisition and position sensing is amplified by the derivative part. For this reason, the PID controllers are seldom used in e-Drives.

4. Hysteresis controller

This controller has a discontinuous action as the output has only two values: maximum or minimum

The hysteresis controller obtains a very good dynamic response, that is better compared to conventional controllers. The current regulation with hysteresis controller can be implemented also for single-phase or three-phase inverters.

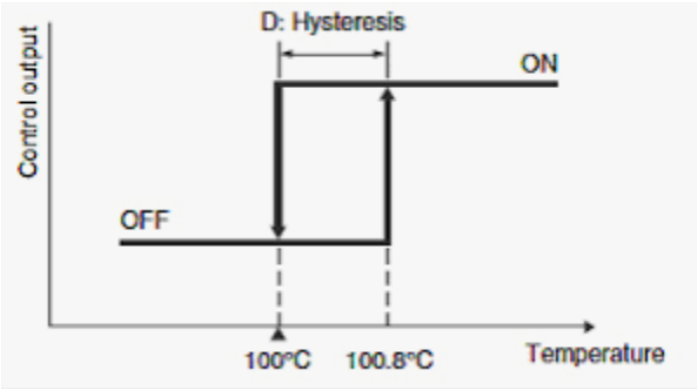


Figure 2.45 – Hysteresis Controller

5. Lead-lag controller (Not very used in e-Drives):

$$H_c(s) = k_p * \frac{s + w_z}{s + w_p}$$

6. Proportional-resonant (P-RES):

$$H_c(s) = k_p + \frac{k_i * s}{s^2 + w_o^2}$$

2.6.4 FOC controller

Vector control, also known as Field Oriented Control (FOC), is a control technique for an AC induction motor (ACIM) and a Permanent Magnet Synchronous Motor (PMSM). The FOC offers good control over the entire torque and speed range.
A transformation of stator currents from the stationary reference frame to the rotor flux reference frame (also known as dq) is required for the implementation of a FOC architecture.
Two of the most used control modes of FOC are Speed control and Torque control while the position control is seldom used:

- The **torque control:** this mode is used for most of the traction applications in which the motor control system follows a reference torque value.
- The **speed control:** in this mode the motor controller follows a reference speed value and generates a torque reference for the torque control that forms an inner subsystem.

The implementation of the FOC control can be done in two different ways:

1. **With Sensor**: It measures current and position by using sensors;
2. S**ensorless**: uses the estimated feedback values instead of the actual sensor based.

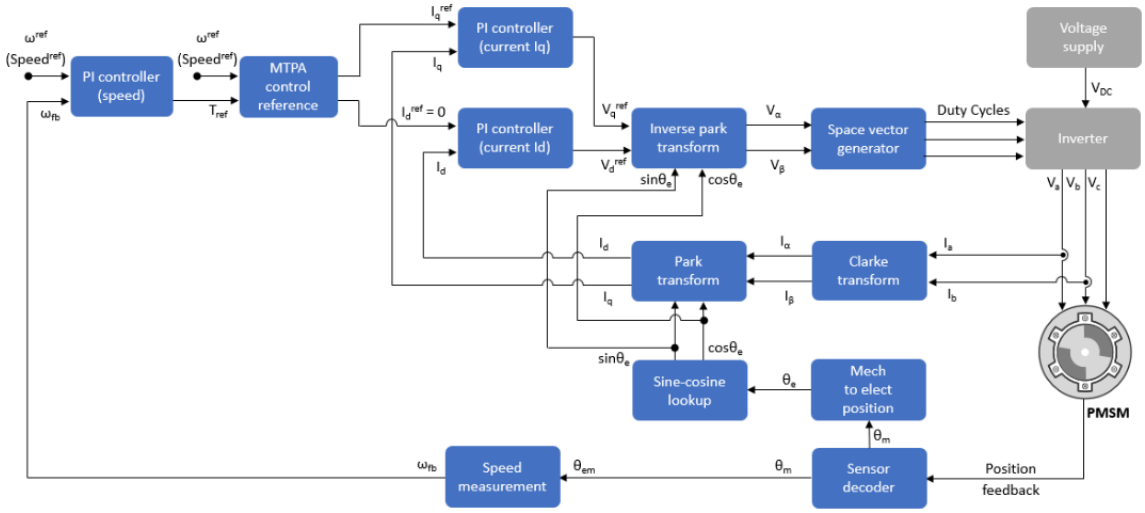


Figure 2.46 – FOC Control

2.7 Inverters

In this chapter we will explain what an inverter is and how it works, which parts it is made of, but above all what are the main types of inverters most used in common situations.

Inverters are widely used in the renewable energy sector and in a large number of devices such as accumulators to automotive electric motors and air conditioners.
An inverter is an electronic device that has the function of transforming a direct current (DC) into an alternating current (AC) at certain voltage and frequency.
Its use is essential for powering electrical devices operating in alternating current via direct current.
Inverter are used in stand-alone photovoltaic systems to power electronic devices in isolated house, mountain huts, campers, boats and they are also used in grid connected photovoltaics in order to enter the current produced by the system directly into the electricity distribution network.
Inverters are also used in many other applications, ranging from uninterruptible power supplies to electric motor speed controllers.

2.7.1 Single-phase half-bridge inverter

This type of inverter is composed by one bidirectional switching cell with symmetrical DC voltage supply $\pm V_{dc}$ with respect to a physical central point 0:

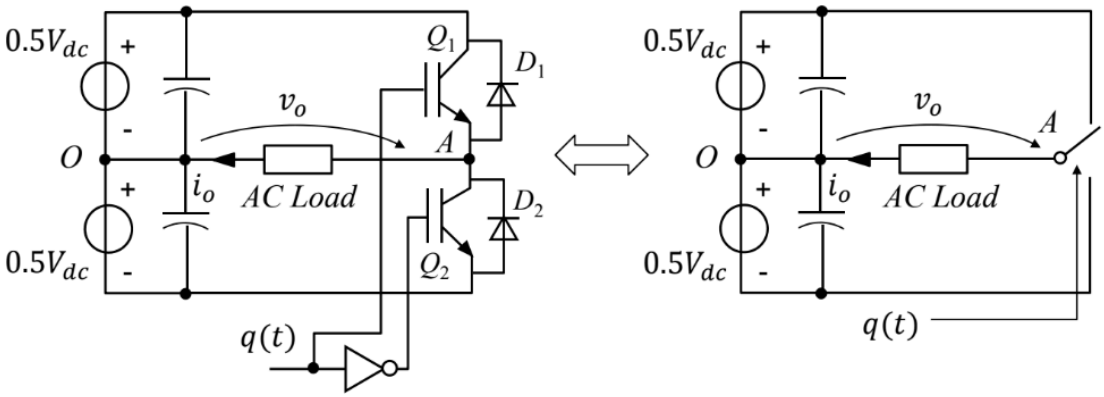


Figure 2.47 – Single-phase half-bridge inverter

The AC load is connected between the cell output A and the central supply point 0. Independently of the sign of the load current and considering an ideal bidirectional switching cell, the output voltage depends only on the voltage supply and the switching function:

$$\begin{cases} q = 1 \rightarrow v_0 = +0.5V_{dc} \\ q = 0 \rightarrow v_0 = -0.5V_{dc} \end{cases} \rightarrow \quad v_0(t) = [2 \cdot q(t) - 1] \cdot \frac{V_{dc}}{2}$$

If the switching function is generated by a PWM block using an unipolar carrier of frequency $f_s = {}^{1}/_{T_s}$:

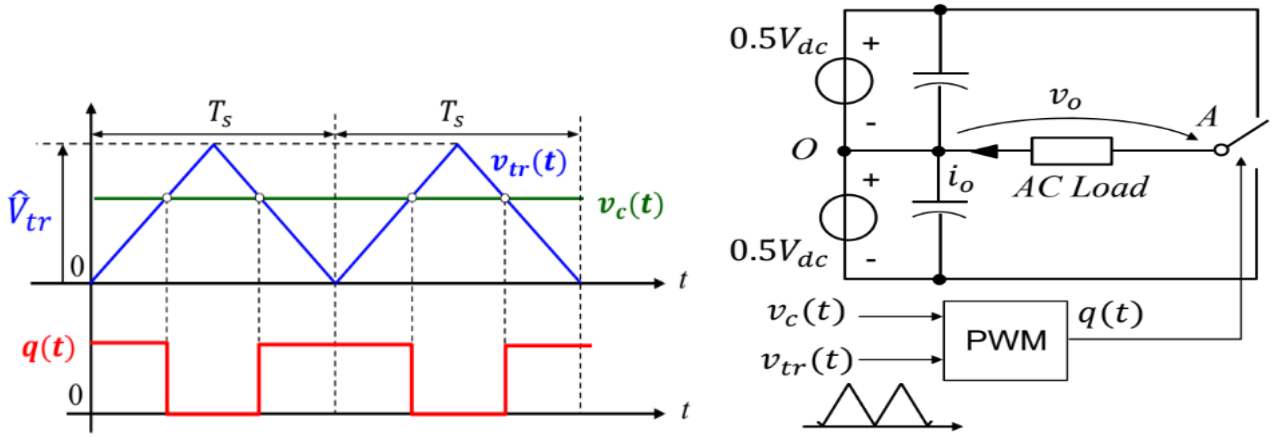


Figure 2.48 – Single-phase half-bridge inverter with PWM switching function

we have that:

$$\begin{matrix} q(t) = 1 \ \ if \ \ v_c(t) > v_{tr}(t) \\ q(t) = 0 \ \ if \ \ v_c(t) < v_{tr}(t) \end{matrix} \rightarrow \quad d(t) = \frac{1}{\hat{V}_{tr}} \cdot v_c(t)$$

The output mobile mean voltage is:

$$\bar{v}_o = \frac{1}{T_s}\int_0^{T_s} v_0(t)dt = \frac{V_{dc}}{2T_s}\cdot\int_0^{T_s}[2\cdot q(t)-1]dt \quad \rightarrow \quad \bar{v}_o = \frac{V_{dc}}{2}\cdot[2\cdot d(t)-1]$$

then, substituting $d(t)$ in the last equation:

$$\bar{v}_o = V_{dc}\cdot\left[\frac{v_c(t)}{\hat{V}_{tr}} - 0.5\right]$$

The inverter must generate a sinusoidal mobile mean value (moving average). So, if the desired output mobile mean value is:

$$\bar{v}_0^* = \hat{V}^*\cdot\sin(\omega t) = V_{dc}\cdot\left[\frac{v_c(t)}{\hat{V}_{tr}} - 0.5\right] \quad \rightarrow \quad v_c(t) = \frac{\hat{V}^*}{V_{dc}}\cdot\hat{V}_{tr}\cdot\sin(\omega t) + 0.5$$

The command v_c must be sinusoidal:

- The offset of 0.5 results from the unipolar PWM modulator
- $\omega = 2\pi f_0, f_0(Hz)$ is the desired output frequency

The half-bridge inverter is a 2-level inverter:

$$\begin{cases} if\ \ v_c(t) > v_{tr}(t) \ \rightarrow\ q = 1 \ \rightarrow\ v_0(t) = +\dfrac{V_{dc}}{2} \\ if\ \ v_c(t) < v_{tr}(t) \ \rightarrow\ q = 0 \ \rightarrow\ v_0(t) = -\dfrac{V_{dc}}{2} \end{cases}$$

The computation of the command from the desired voltage:

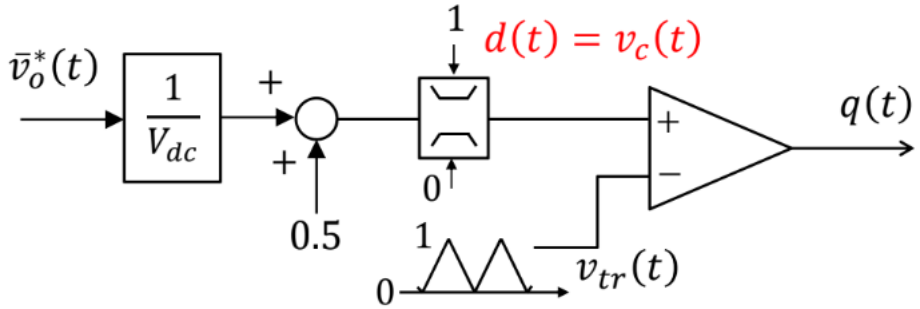


Figure 2.49 – PWM Command block

- The term $\frac{\bar{v}_0^*}{V_{dc}} + 0.5$ must be saturated between 0 and 1 since the command must be in the range $[0, \hat{V}_{tr}]$;
- $\frac{\bar{v}_0^*}{V_{dc}} = \bar{v}_{0,norm}^*$ is the normalized desired voltage with respect to V_{dc}.

The analysis of the inverter can be performed also with a PWM block that uses a carrier $v_{tr}(t)$ having unity peak value, then the command $v_c(t)$ becomes the cell duty-cycle:

$$d(t) = v_c(t) \;\rightarrow\; \bar{v}_0^*(t) \in \left[-\frac{V_{dc}}{2}, +\frac{V_{dc}}{2}\right]$$

we can than define, respectively, the amplitude and frequency modulation index:

$$m_a = \frac{\hat{V}^*}{0.5 \cdot V_{dc}} \quad , \quad m_f = \frac{f_s}{f_o}$$

Generally, inverters operate in function of the amplitude modulation index

1. *Sinusoidal linear modulation* $\left(m_a \leq 1 \;\&\; \hat{V}^* \leq \frac{V_{dc}}{2}\right)$:

   The fundamental $v_{o,1}$ voltage is the mobile mean value if the frequency modulation index is high enough $(m_f \geq 20)$:

   $$\bar{v}_o(t) \cong v_{o,1}(t) = \hat{V}^* \cdot \sin(2\pi f_o t)$$

   The voltage harmonics are grouped in sidebands around switching frequency and multiples; the load inductance is able to filter the load current that is almost sinusoidal (excepting the switching ripple).
   This type of modulation is linear because the peak of the fundamental is equal to the peak of the reference voltage (ideal inverter).
   The figure that follows, shows an example of a sinusoidal linear modulation:

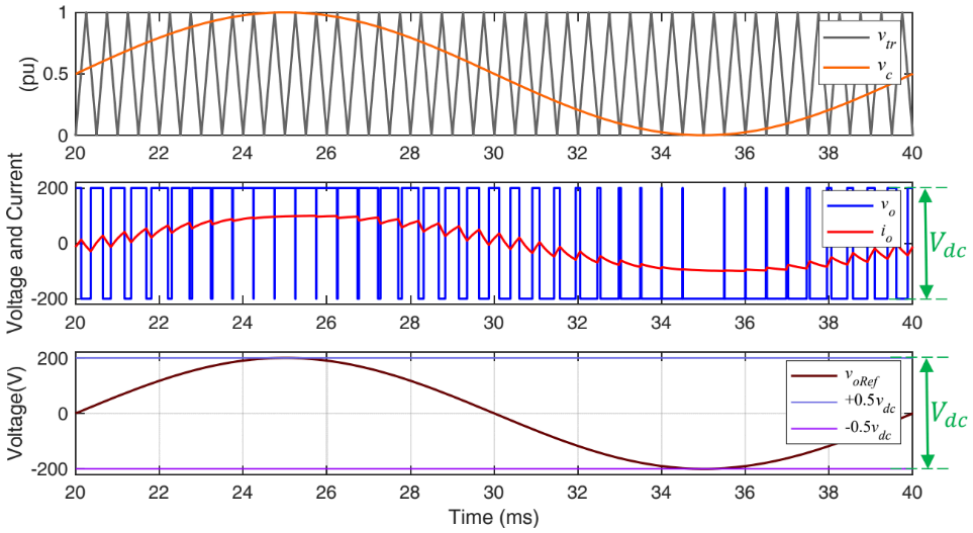


Figure 2.50 – Sinusoidal linear modulation

2. *Overmodulation* $\left(m_a > 1 \ \& \ \hat{V}^* > \frac{V_{dc}}{2}\right)$

It is used to provide an output voltage with a fundamental component higher than $0.353 \cdot V_{dc}$. Then, the duty-cycle is saturated between 0 and 1 (it starts becoming a trapezoidal waveform).

During the saturation of duty-cycle, the cell does not switch, in this way the instantaneous output voltage is clamped at $-\frac{V_{dc}}{2}$ or at $+\frac{V_{dc}}{2}$.

The figure that follows, shows an example of an overmodulation:

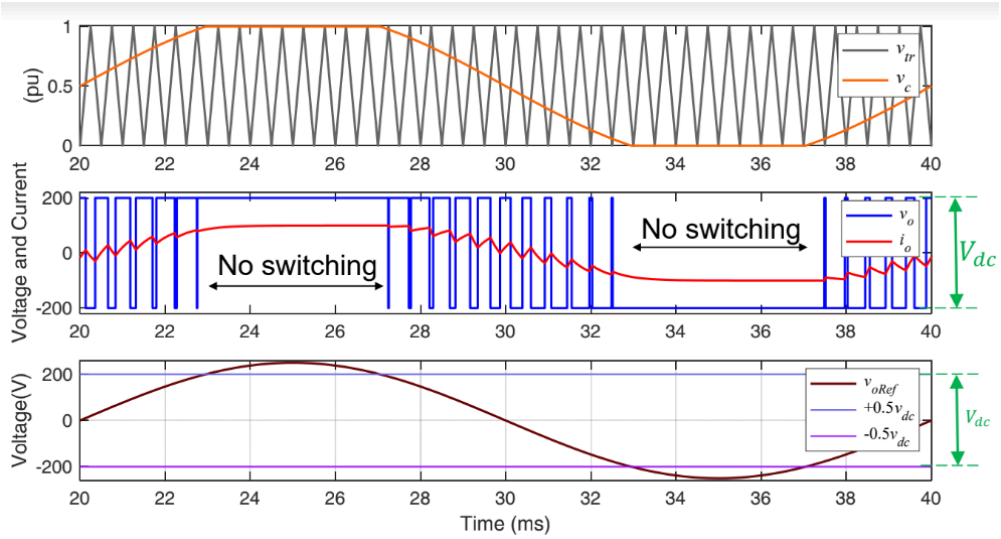


Figure 2.51 – Overmodulation

The fundamental $v_{o,1}$ voltage for overmodulation has lost his linearity due to overmodulation:

$$V_{o,1} < m_a \cdot \frac{V_{dc}}{2\sqrt{2}}(V)$$

3. *Square wave operation* $(m_a \gg 1)$:

   The duty-cycle becomes a square wave, in this way the output voltage is consequently a square wave of amplitude $\frac{V_{dc}}{2}$ with only two commutations over the electrical period $\frac{1}{f_o}$.
   The fundamental $v_{o,1}$ voltage reaches its maximum value:

$$V_{o,1}, max = \frac{4}{\pi} \cdot \frac{V_{dc}}{2\sqrt{2}} = 0.45 V_{dc}(V)$$

The figure that follows, shows an example of a square wave modulation:

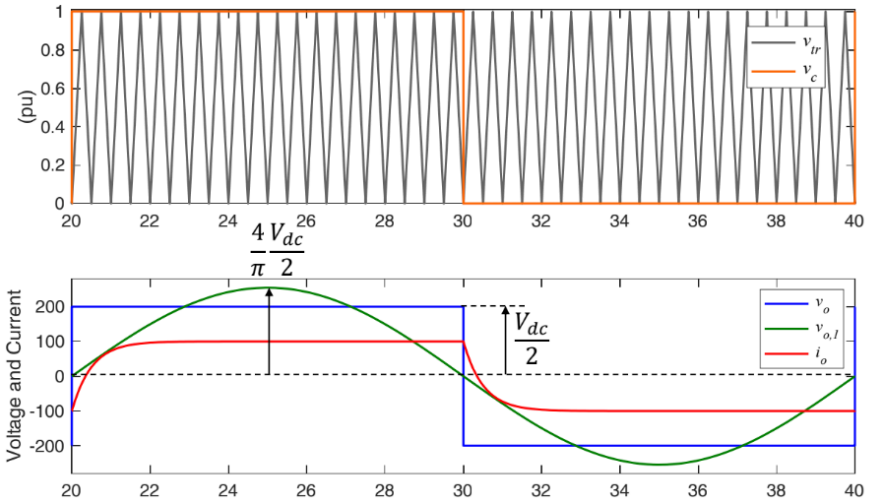


Figure 2.52 – Square wave operation

2.7.2 Single-phase full-bridge inverter

A full bridge is obtained with a single phase PWM inverter with two bidirectional switching cells and a single DC voltage supply, with the load that is connected between the outputs A and B of the two switching cells:

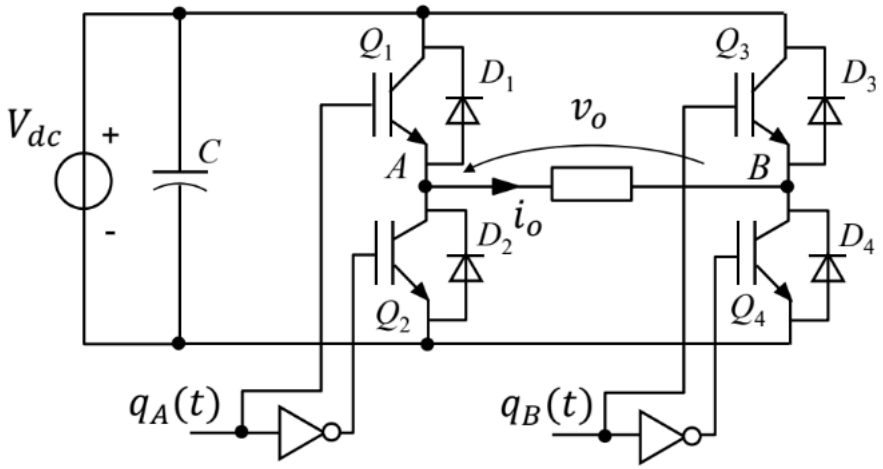


Figure 2.53 – Single-phase full-bridge inverter

The PWM modulator uses a single command v_c and unipolar carrier v_{tr} having the amplitude equal to unity.
In general, we can distinguish three types of PWM modulation techniques.

1. *Bipolar PWM modulation techniques*

$$\begin{cases} if\ \ v_c(t) > v_{tr}(t) \ \rightarrow\ q_A = 1 \ \rightarrow\ v_0(t) = +V_{dc} \\ if\ \ v_c(t) < v_{tr}(t) \ \rightarrow\ q_A = 0 \ \rightarrow\ v_0(t) = -V_{dc} \end{cases}$$

then:

$$v_0(t) = [2 \cdot q_A(t) - 1] \cdot V_{dc}$$

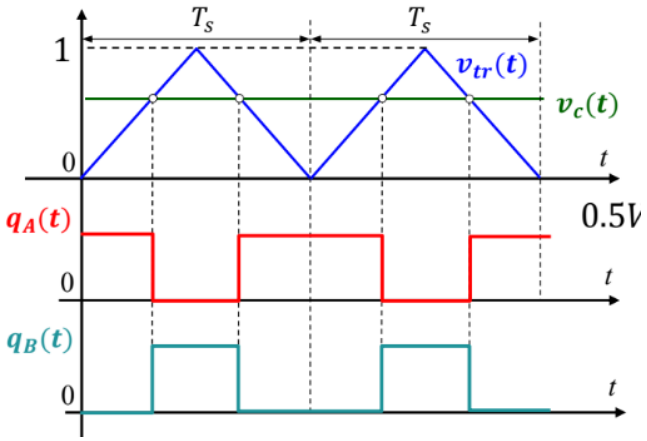


Figure 2.54 – Bipolar PWM

The mobile mean value of the output voltage is:

$$\bar{v}_o = \frac{1}{T_s}\int_0^{T_s} v_0(t)dt = \frac{V_{dc}}{T_s}\cdot\int_0^{T_s}[2\cdot q_A(t)-1]dt \quad \rightarrow \quad \bar{v}_o(t) = V_{dc}\cdot[2\cdot v_c(t)-1]$$

Inverting the last formula, we obtain that the PWM command $v_c(t)$ becomes the duty-cycle of cell A:

$$v_c(t) = \frac{\bar{v}_o(t)}{2\cdot V_{dc}} + 0.5 \;\rightarrow\quad v_c(t) = d_A(t) = 1 - d_B(t)$$

2. <u>Unipolar PWM modulation</u>

This type of modulation has an independent switching operation between the cell A and B:

$$\begin{cases} q_k = 1 \quad \rightarrow \quad v_{k0} = +\dfrac{V_{dc}}{2} \\ q_k = 0 \quad \rightarrow \quad v_{k0} = +\dfrac{V_{dc}}{2} \end{cases}, \qquad k = A, B$$

Obtaining that:

$$v_0(t) = [q_A(t) - q_B(t)]\cdot V_{dc}$$

The mobile mean value of the output voltage is:

$$\bar{v}_o = \frac{1}{T_s}\int_0^{T_s} v_0(t)dt = \frac{V_{dc}}{T_s}\cdot\int_0^{T_s}[q_A(t)-q_B(t)]dt \quad \rightarrow \quad \bar{v}_o(t) = V_{dc}\cdot[2\cdot v_{cA}(t)-1]$$

Inverting the last formula, we obtain that the PWM command $v_c(t)$ becomes:

$$v_{cA}(t) = \frac{\bar{v}_o(t)}{2\cdot V_{dc}} + 0.5, v_{cB}(t) = -\frac{\bar{v}_o(t)}{2\cdot V_{dc}} + 0.5 = 1 - v_{cA}(t)$$

3. <u>Phase shift (PS):</u>

It represents a particular square wave operation that allows the regulation of the fundamental voltage. The duty-cycle for the two cells are $d_A = d_B = 0.5$ and the switching function $q_A(t)$ and $q_B(t)$ are phase-shifted by Φ_{PS}:

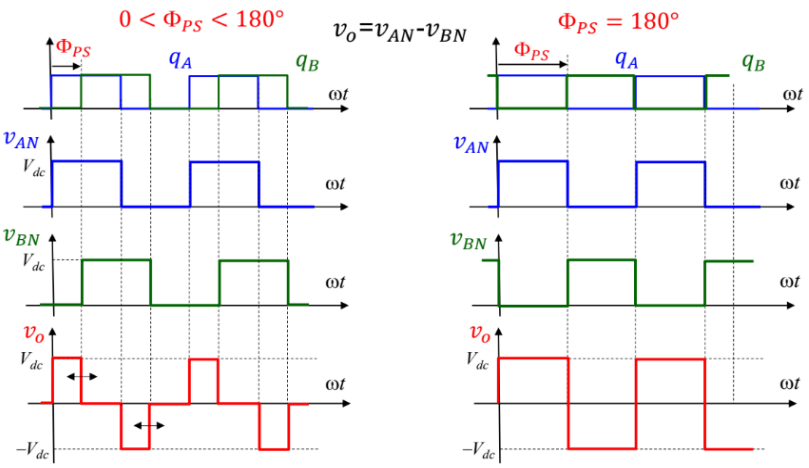


Figure 2.55 – Phase shift

2.7.3 Three-phase inverter

The three-phase inverter differs from the single-phase one because the phase switching functions are generated by a PWM block having three reference voltages and using a single carrier.

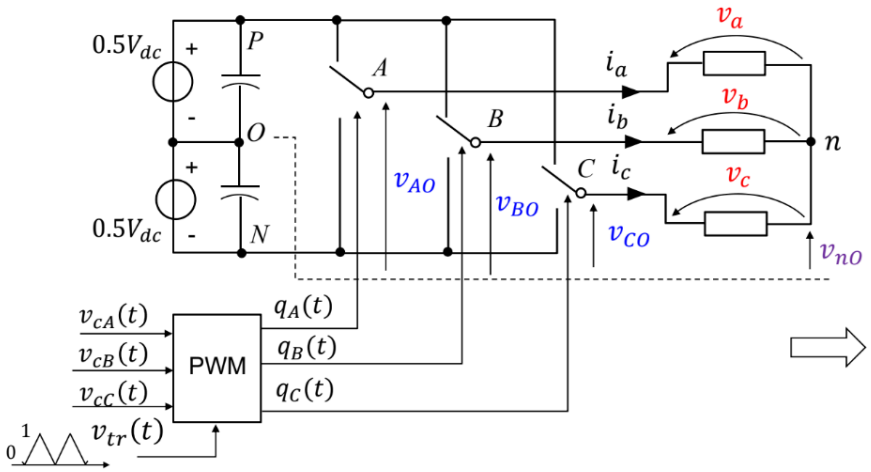


Figure 2.56 – Three-phase inverter

Sinusoidal PWM, generation of symmetrical three-phase voltages using a symmetrical three-phase system of commands:

$$v_{cA}(t) = d_A(t) = \frac{\hat{V}^*}{V_{dc}} \cdot \sin(\omega t) + 0.5$$

$$v_{cB}(t) = d_B(t) = \frac{\hat{V}^*}{V_{dc}} \cdot \sin\left(\omega t - \frac{2\pi}{3}\right) + 0.5$$

$$v_{cC}(t) = d_C(t) = \frac{\hat{V}^*}{V_{dc}} \cdot \sin\left(\omega t + \frac{2\pi}{3}\right) + 0.5$$

$$\bar{v}_{ko} = \frac{V_{dc}}{2} \cdot [2 \cdot v_{ck}(t) - 1], k = A, B, C$$

$$\rightarrow \quad \bar{v}_{AO} = \hat{V}^* \cdot \sin(\omega t)$$

$$\bar{v}_{BO} = \hat{V}^* \cdot \sin\left(\omega t - \frac{2\pi}{3}\right)$$

$$\bar{v}_{CO} = \hat{V}^* \cdot \sin\left(\omega t + \frac{2\pi}{3}\right)$$

The inverter imposes a symmetrical three-phase voltage system with respect to the DC link midpoint.
The link between the inverter voltages and the phase load voltage is the common mode voltage $v_{nO}(t)$:

$$v_a(t) = v_{AO}(t) - v_{nO}(t)$$
$$v_b(t) = v_{BO}(t) - v_{nO}(t)$$
$$v_c(t) = v_{CO}(t) - v_{nO}(t)$$

Where in general, for a n-phase system, the common voltage is:

$$v_{nO}(t) = \frac{1}{3} \cdot (v_{AO} + v_{BO} + v_{CO} + \cdots)$$

but $\bar{v}_{nO}(t) = 0$ so we obtain that:

$$\bar{v}_a(t) = \bar{v}_{AO}(t)$$
$$\bar{v}_b(t) = \bar{v}_{BO}(t)$$
$$\bar{v}_c(t) = \bar{v}_{CO}(t)$$

The inverter operation in function of the amplitude modulation index is the same as for the single-phase half-bridge inverter (Sinusoidal linear, modulation, Overmodulation and square wave operation).

2.8 Turbocharger

Turbochargers are typically implemented in engines as a technological step towards engine downsizing.
In a conventional turbocharger, the turbine is driven by the exhaust gases from the engine cylinder, and the air is loaded into the cylinder inlet when the impeller connected to the turbine is rotated.

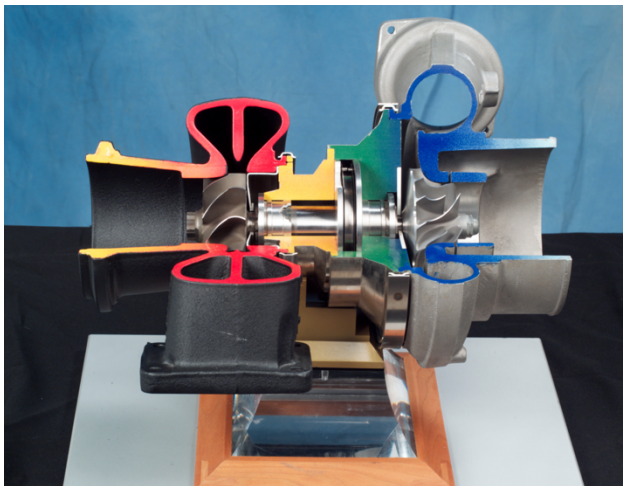


Figure 2.57 - Turbocharger

One factor of vehicle performance degradation is turbo lag caused by low engine revolutions per minute. A low engine speed results in a low exhaust gas speed and a correspondingly low rotational speed of the turbine, which reduces the supercharging capacity of the impeller.
To reduce this turbo lag, and thus improve vehicle performance and drivability, the answer is a turbocharger system assisted by a very high-speed electric motor.
Among the very high-speed compact motors, permanent magnet synchronous motors (PMSM) are considered the best. Their high-power density leads to a high response speed. Furthermore, the non-electric excitation in the rotors produces excellent efficiency.
If the speed response increases, the impeller reaches the rated speed much sooner and this leads to an improvement in performance for the removal of turbo lag. Consequently, meeting the required specifications and speed response characteristics are significant problems in the design of very high-speed machines.

3. IMPLEMENTATIVE TOOLS

In this chapter an overview will be made of all the implementation tools that have been used during my thesis path in the company and that were useful to achieve my final goal.

3.1 Matlab

MATLAB (also known as Matrix Laboratory) is an environment for numerical computation and statistical analysis written in C, which also includes the programming language of the same name created by MathWorks.

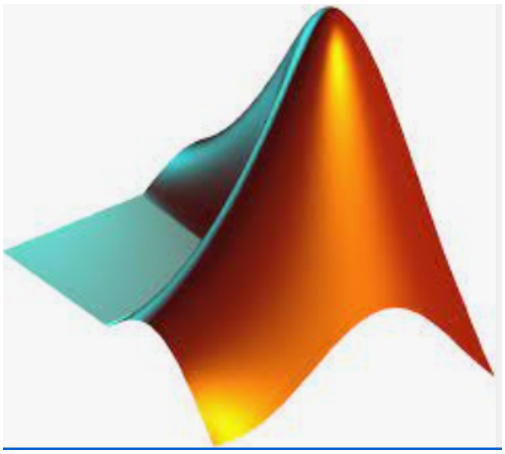


Figure 3.1 - Matlab

MATLAB allows you to manipulate matrices, visualize functions and data, implement algorithms, create user interfaces, and interface with other programs.

MATLAB was created in the late 1970s by Cleve Moler, president of the computer science department of the University of New Mexico.

The main MATLAB interface consists of several windows that you can tile, move, minimize, resize, and so on.

The main and most widely used windows are the following four:

- Command window: is a window of the main MATLAB interface, in which it is possible to type supported commands and view the results on the screen in real time;
- Workspace: is the workspace (or memory space) containing the declared variables;
- Current directory: it allows to explore the contents of the folders on your memory medium;
- Command history: all recently typed commands are listed, divided by time and date.

3.2 Simulink

Simulink is a software for modeling, simulation and analysis of dynamic systems, developed by the US company MathWorks and it is tightly integrated with MATLAB. It is a block diagram environment used to design systems with multidomain models, perform simulations before moving on to hardware and proceed with distribution without writing code.
Simulink is used for Model-Based design, for simulation, for Model-Based System Engineering, for agile software development.
The advantages of Simulink are that it requires neither the formulation of differential equations, nor the knowledge of a particular language. It uses a graphical interface that allows you to build models such as block diagrams. It provides users with a wide range of predefined functional blocks, so that almost all projects are reduced in practice to the appropriate interconnection of these blocks.
Simulink contains a library of blocks that describes elementary static and dynamic elements. The user composes the block diagram of the system to be simulated by interconnecting the elementary blocks. Simulink automatically generates the equations and solves the desired numerical simulation problem.
Models built in Simulink can be hierarchical models: each block of the system can itself be a complex subsystem.
Simulink interacts with MATLAB through the Workspace (Simulink models can contain Workspace variables)
Similarly, the result of the simulations can be exported to the Workspace and analyzed with MATLAB.

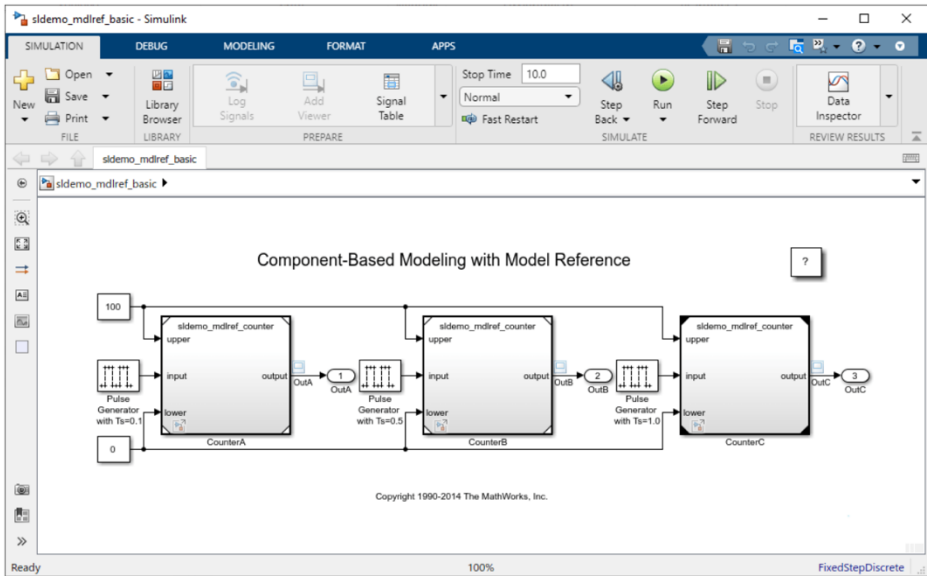


Figure 3.2 – Simulink

3.3 System generator for DSP(XSG)

Xilinx System Generator for DSP (SysGen) is a MATLAB Simulink add-on that allows the development of FPGA architectures using block graphics programming.

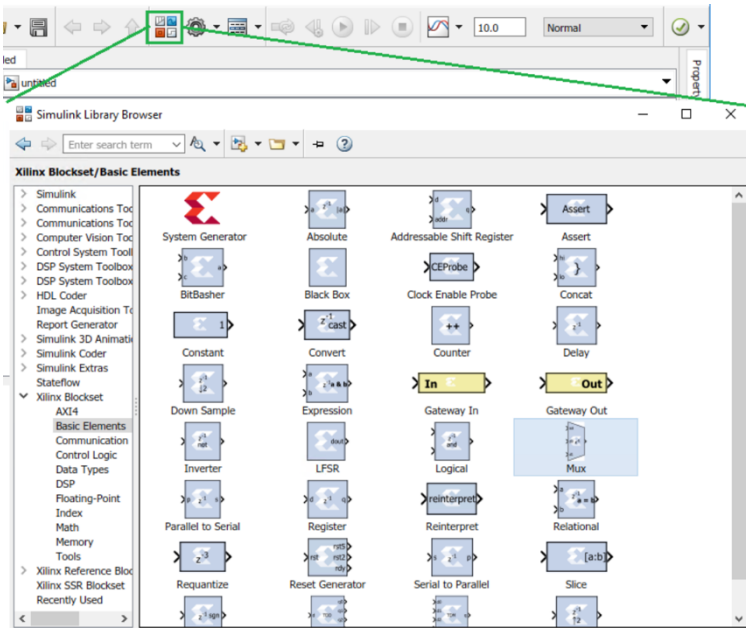


Figure 3.3 System generator for DSP

Users can validate their designs via simulation in Simulink and the design can be packaged into a Vivado IP and easily imported into a Vivado project.

The logic of the FPGA must be placed in a subsystem. The Gateway In and Gateway Out blocks are used to represent the FPGA's input / output ports. Between the Gateway blocks there can only be blocks that come from the Xilinx Toolbox / HDL library. We need to add the System Generator block as a control panel for simulation and code generation.

3.4 Configuration Desk

Configuration desk is a very intuitive graphical configuration and implementation software, both for large hardware-in-the-loop (HIL) tests and for the management of RCP (Rapid control prototyping) applications. They are based on dSpace real-time hardware such as SCALEXIO, the simulator used in our thesis, which includes the implementation of behavioral models and I/O function code.

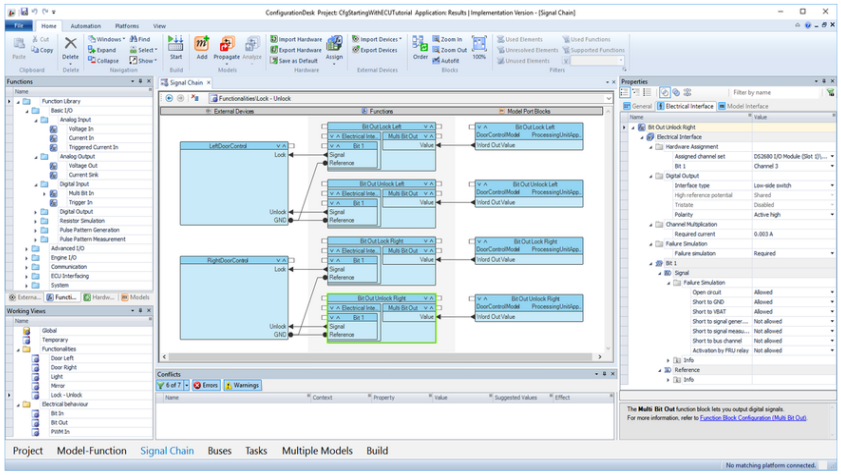


Figure 3.4 - Configuration Desk

Configuration Desk from a general structure of external devices, configures real-time hardware and the connected behavioral model.

Working with Configuration Desk offers a wide range of benefits when developing and testing controllers:

- It allows you to completely manage the signal path from the external device to the model interface, giving a clear and clean view of the entire application;
- Separate the behavioral model from the I / O port configuration making the model flexible and reusable;
- Simulate I / O with fixed values allowing you to test even if the real I / O are not yet available;
- Allows the automatic implementation of your application on Real-Time dSPACE hardware;
- An intuitive graphical display guided through the workflow for CPR and HIL applications;
- The use of multiple models allows you to build large modular applications.

With MIPS and Simulink Coder it is possible to generate Simulink Implementation Container (SIC) files based on Simulink models.
These SIC files include all the codes needed to run the models in different projects and on different dSPACE platforms (such as VEOS, MicroAutoBox III and SCALEXIO).
This code will be loaded on a real-time application through the use of ConfigurationDesk.

3.5 Control Desk

Control Desk is a dSPACE software used for the development of the control units. It allows you to carry out all the tasks from a single work environment, from the beginning of the experiment to the end. It is used for Rapid Control Prototyping (RCP), for Hardware in the loop (HIL) simulations, for ECU measurements, calibration and diagnostics, allows access to system buses such as CAN, CAN FD, LIN and Ethernet and allows virtual validation with VEOS and SCALEXIO.

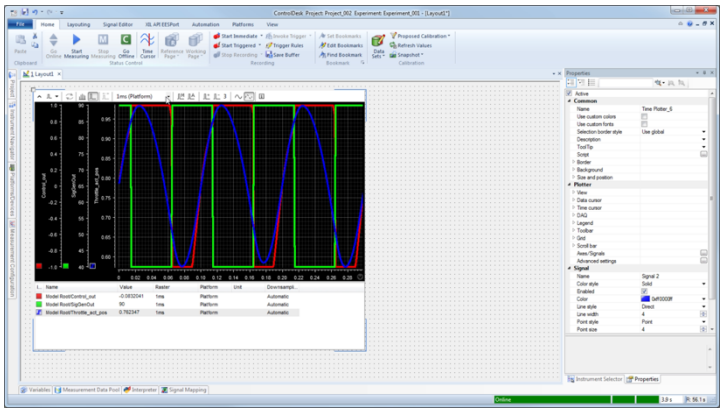


Figure 3.5 - Control Desk

With ControlDesk you can prepare project / experiment data that can be used later in operator mode.

Through ControlDesk it is possible to control real-time applications developed in the Simulink environment and configured / implemented in ConfigurationDesk.
It allows you to create a graphical interface to control and view all the variables of interest in the model.

4. HARDWARE

In this new chapter we will see in detail all the hardware components that were invaluable to complete the thesis. In particular, we will introduce what is the control unit used to close the 'loop' and used to test the model of the PMSM motor and the inverter.
Finally, the SCALEXIO simulator and the DS2655 board (FPGA) used to carry out the tests and obtain the desired results will be illustrated.

4.1 Controller

At the beginning, the purpose of the thesis was to test an external control unit that received the currents in phase I_{abc} from the outside and that provided the duty-cycles of each phase of the motor at the output, and consequently at the input to the model to be tested.
Subsequently, in the absence of this control unit, it was decided to deviate towards a different but still effective solution. In reality, the control unit used to test the model is a controller model found in the *dSpace* libraries. In particular, this model is a microcontroller where the software used to implement the sensorless field-oriented control (FOC) is loaded, therefore it does not receive the speed/position feedback but calculates it internally through an algorithm.
The dSpace controller model is divided into a part that runs on the processor and a part that runs in FPGA:

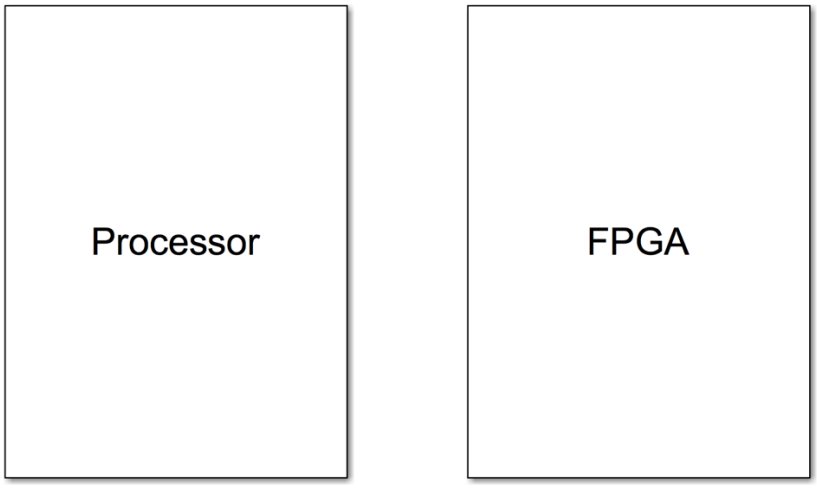


Figure 4.1 – Controller model structure

The main part of the model that runs on the Processor uses an interrupt generator from the FPGA, which is activated when the FPGA finishes the calculation:

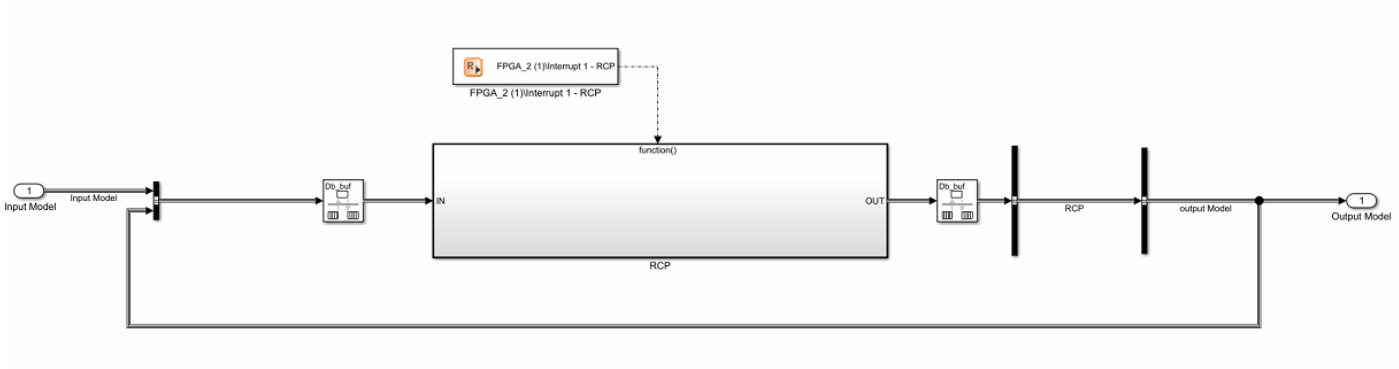


Figure 4.2 – Processor part of controller

The RCP block is divided into 3 subsystems:

- ACTUATOR;
- SENSOR;
- CONTROLLER.

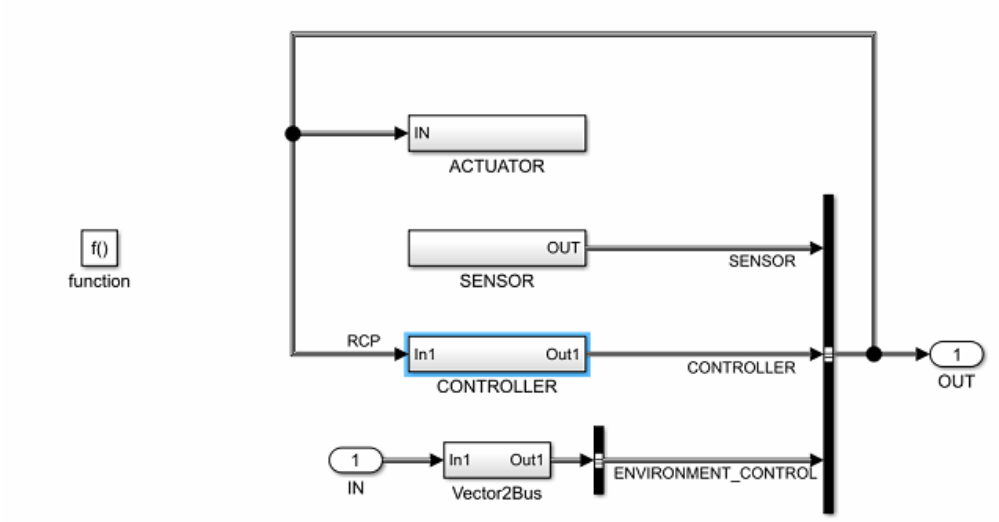


Figure 4.3 – RCP Block

In addition to the three blocks mentioned above, there is the addition of the 'Vectorbus' block used by the library to manage inputs from the outside, in particular from the electric motor model.

The FPGA model, on the other hand, is composed of three subsets:

- ACTUATOR;
- SENSOR;
- INTERRUPT.

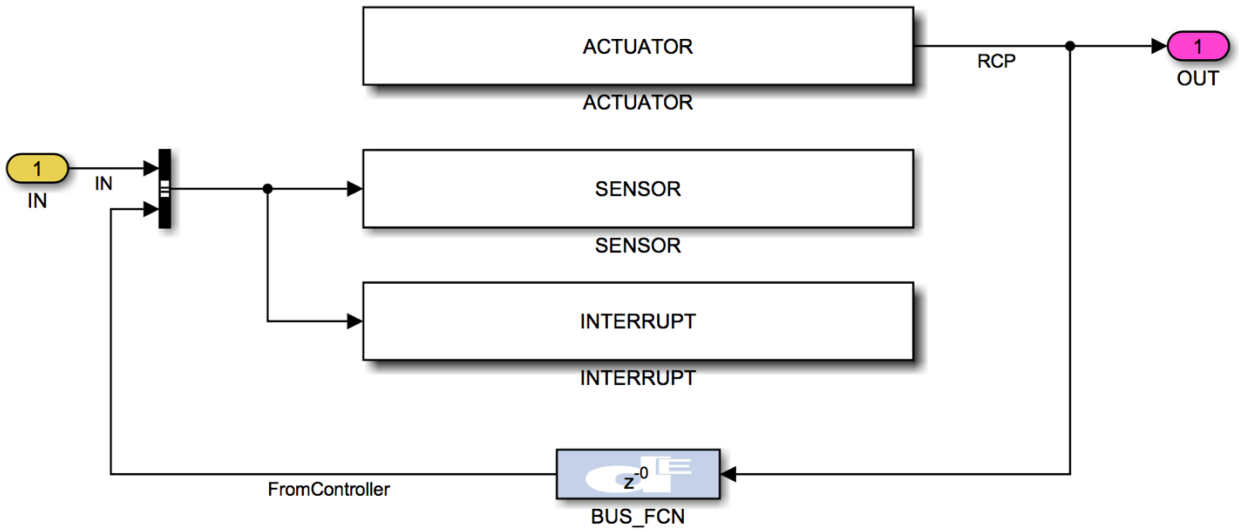


Figure 4.4 – FPGA part of the controller

This model receives in input the three currents in phase I_{abc}, which arrive from the motor model:

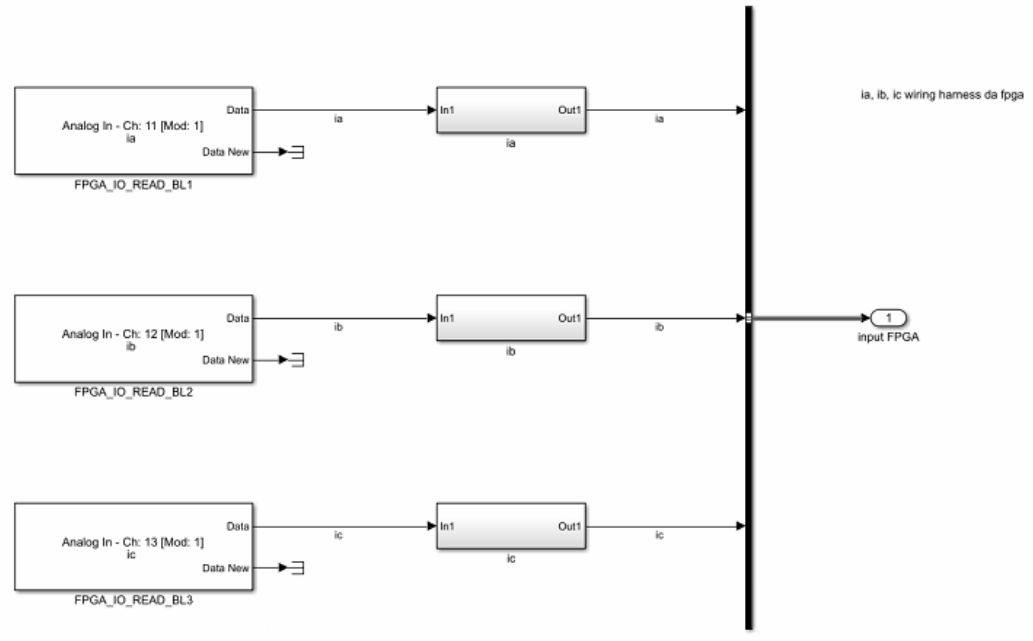


Figure 4.5 – Controller input

These signals subsequently enter in the SENSOR block where the average current for each phase is calculated. Then, this value is sent to the processor where the FOC control logic is implemented.
In addition to the averaged phase currents, the processor has as input the supply voltage, the mechanical angle, the mechanical speed and the target speed, plus other inputs used in the model as set point for open loop control in current or voltage, which they will not be used.

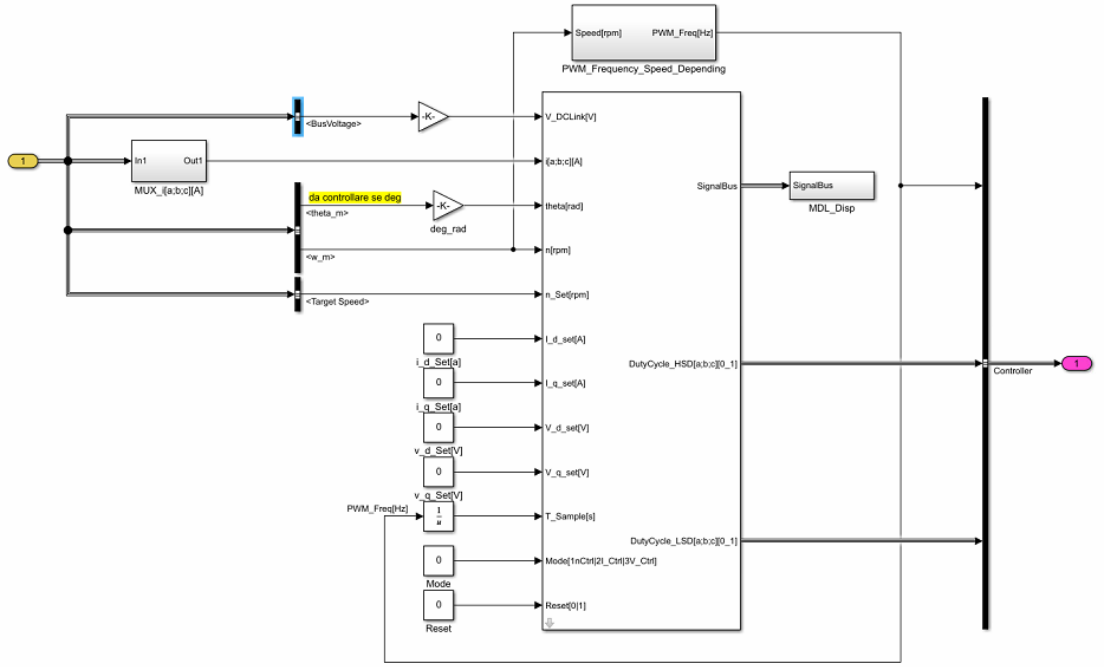


Figure 4.6 – Sensor block

At the output of the CONTROLLER block there are 6 duty cycles to control the inverter high and low switches used by the ACTUATOR block, present in the FPGA part, to generate the output PWMs

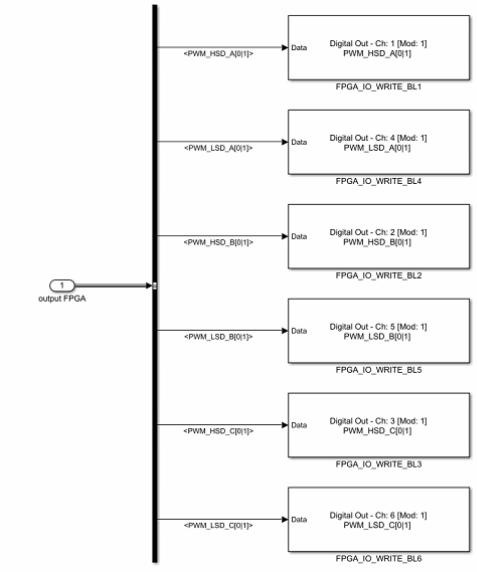


Figure 4.7 – Controller output

4.2 SCALEXIO PROCESSING UNIT

The SCALEXIO Processing Unit are based on an industrial PC with an Intel XEON processor, a real-time operating system, and an IOCNET plug-on card for communication with the I/O and with other real-time processors.

Figure 2.8 - Scalexio

A processing Unit can be equipped with four or eight IOCNET ports.

You can use IOCNET ports to connect multiple SCALEXIO Processing Units, to set up multiprocessor systems or to couple a DS100x-based system to a SCALEXIO Processing Unit via Gigalink. One of the available processor cores is reserved for system services, the others can be used for computing real-time models.

Below are the technical data of the SCALEXIO Processing Units:

Parameter		Specification	
		High Core Performance	High Parallel Performance
Processor		■ Intel XEON® E3-1275v6 ■ Frequency: 3.8 GHz ■ Number of cores: 4 ■ 3 cores for model computation ■ 1 core for services, e.g., host communication	■ Intel® XEON® Gold 6208U ■ Frequency: 2.9 GHz ■ Number of cores: 16 ■ 15 cores for model computation ■ 1 core for services, e.g., host communication
Memory		■ L1 cache: 32 + 32 kB (data + instructions) ■ L2 cache: 256 kB ■ L3 cache: 8 MB ■ 16 GB RAM	■ L1 cache: 32 + 32 kB (data + instructions) ■ L2 cache: 1 MB ■ L3 cache: 22 MB ■ 64 GB RAM
Solid State Disk (SSD)		■ Data recording and replay ■ Capacity: 480 GB ■ Optional	■ Data recording and replay ■ Capacity: 480 GB ■ Included
Angular processing unit (APU)		■ 6 APUs on the DS2502 IOCNET Link Board	
	Angular resolution	■ 0.011°	
	Speed range	■ ± 28,610 rpm	
	Speed resolution	■ 0.109 rpm	
Interfaces	IOCNET	■ 4 or 8 IOCNET connectors on the DS2502 IOCNET Link Board (can be optionally used as Gigalink connectors)	
	PCI Express/PCI[1]	■ 2x PCI ■ 5x PCIe (2x x16, 1x x8, 2x x4)	■ 2 x PCIe x16 slot ■ 3 x PCIe x8 slot ■ 1 x PCIe x4 slot ■ 1 x PCIe x1 slot
	Ethernet interface	■ Integrated low-latency Gigabit Ethernet I/O interface ■ Additional ports via SCALEXIO Ethernet Boards	■ 2 x Integrated low-latency Gigabit Ethernet I/O interface ■ 1 x 10 Gigabit Ethernet I/O interface ■ Additional ports via SCALEXIO Ethernet Boards
	Serial interface	■ RS232 interface with standard UART allowing transfer rates up to 480.6 kBaud	
Host interface		■ Gigabit Ethernet	■ 10 Gigabit Ethernet
Multiprocessor system		■ Building multiprocessor systems with more SCALEXIO Processing Units and/or processor boards	
Cooling		■ Active cooling	
Ambient temperature		■ Operating temperature 0 °C ... 40 °C (32 °F ... 104 °F)	
Operating humidity		■ 5% ... 95% (non-condensing environment)	
Size (width x height x depth)		■ Rack-mount version: 483 x 132 x 400 mm (19.0 x 5.2 x 15.7 in) ■ Desktop version: 428 x 132 x 400 mm (16.9 x 5.2 x 15.7 in) ■ 19-inch subrack ■ 3 height unit (H)	
Mass		■ Approx. 12 kg	
Power supply		■ 100 ... 240 V AC, 50/60 Hz, 500 W	■ 100 ... 240 V AC, 50/60 Hz, 1200 W

Figure 4.9 – SCALEXIO Technical data

4.3 DS2680 – I/O UNIT

The DS2680 I/O Unit is a MultiCompact I/O unit for the SCALEXIO system that provides all the I/O channels required for the hardware-in-the-loop simulation. It provides 38 I/O functions that are supported by different I/O channels, which are defined and configured graphically with dSpace Configuration desk

Figure 4.10 - DS2680 I/O Unit

Below are shown the several functions and some technical details:

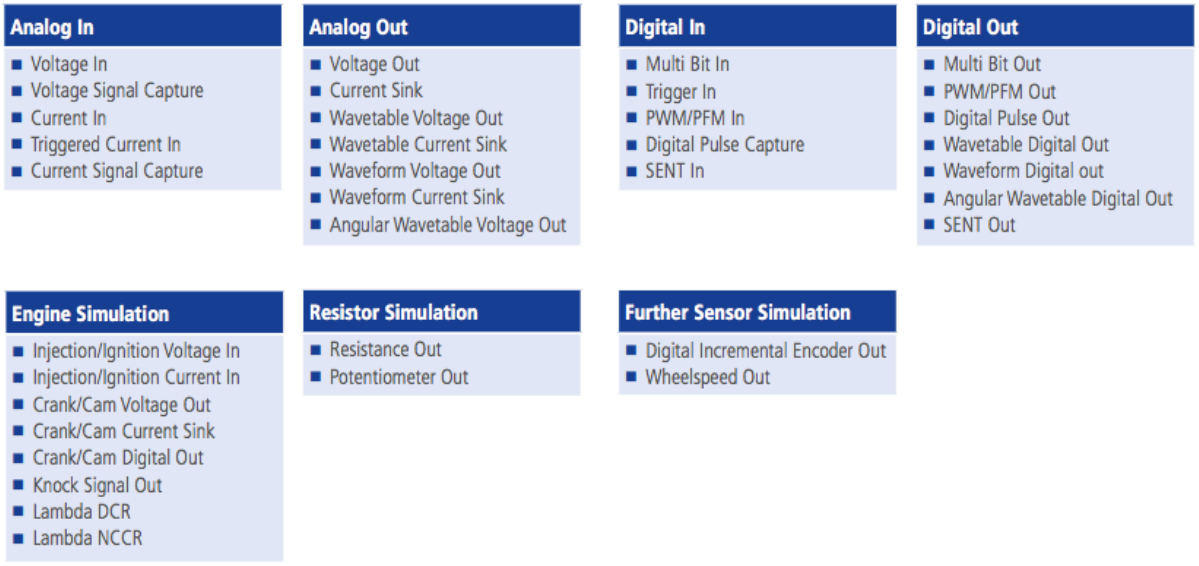


Figure 4.11 - DS2680 I/O function

Parameters		Specification
Signal measurement		■ Max. 6 A per channel ■ Substitute loads pluggable, up to 2 W per measurement channel ■ Connector for real loads ■ Channel bundling to increase current carrying capacity ■ Multifuse for electrical safety
	■ 20 analog inputs (Analog In 1)	■ Only voltage measurement ■ Measurement range 0 ... 60 V ■ Resolution 16 bit
	■ 30 digital inputs (Digital In 1)	■ Voltage measurement ■ Trigger value 0 ... 24 V ■ Voltage range 0 ... 60 V
	■ 18 variable inputs (Flexible In 2)	■ Voltage measurement digital ■ Voltage range 0 ... 60 V ■ Trigger value 0 ... 24 V ■ Current measurement analog and digital ■ Measurement range ±18 A ■ Resolution 16 bit (analog)
Signal generation	■ 15 analog outputs (DC) (Analog Out 1)	■ Only voltage generation ■ Output voltage 0 ... 10 V ■ Output current -5 ... +5 mA ■ Resolution 14 bit
	■ 8 analog outputs (DC) (Analog Out 4)	■ Output voltage 0 ... 10 V ■ Output current -5 ... +5 mA ■ Resolution 14 bit ■ Current sink: Current range -30 ... +30 mA
	■ 7 analog outputs (AC) (Analog Out 3)	■ Only voltage generation ■ Output voltage -20 ... +20 V ■ Resolution 14 bit ■ Effective inner resistance 250 Ω
	■ 12 resistance simulation channels (Resistance Out 1)	■ Resistance range 16 Ω ... 1 MΩ ■ Voltage range -3 ... +18 V to GND ■ Current range -80 ... +80 mA ■ Power max. 250 mW
	■ 28 digital outputs (Digital Out 1)	■ Configurable as low-side/high-side switch or push/pull ■ Low-side = GND ■ High-Side = V_{BAT} or Dig-Out-Ref ■ High-side voltage range: 5 ... 60 V ■ Current range -80 ... +80 mA
Special I/O channels	■ 2 analog input and output groups, e.g., for lambda probe simulation (Analog In 2, Analog Out 2, Load 1)	■ 1 ADC ■ 1 DAC ■ 1 load (component channel) ■ Voltage range -10 ... +10 V ■ Current range -5 ... +5 mA
Voltage supply	■ 1 channel to control the power unit (Power Control 1)	■ Control of TDK-Lambda Genesys™ power supply
	■ 6 power switches without current measurement (Power Switch 2)	■ Up to 60 V ■ Continuous current 4 x 6 A per channel (but a maximum total of 50 A for all channels)

Figure 4.12 - DS2680 I/O technical data

4.4 DS2655 – FPGA BASE BOARD

The DS2655 FPGA Base Board has been designed for applications that require very fast, high resolution signal processing like:

- Electric vehicle applications
- Industrial drive applications
- Electric power industry applications
- Electric drive simulation
- Power electronics simulation
- Power hardware-in-the-loop simulation
- Electric motor control development
- Power electronics control development

Figure 4.13 - DS2655

Three modules are available to expand the I/O channels of the board and up to five I/O modules can be connected to the board, thus providing a flexible, customized channel set.
Here are the technical details:

Parameter	Specification	
	DS2655 7K160	DS2655 7K410
General	■ User-programmable FPGA	
FPGA	■ Xilinx® Kintex®-7 160T ■ Logic cells: 162,240 (DSP slices: 600) ■ Distributed RAM: 2,188 kbit ■ Block RAM: 11,700 kbit	■ Xilinx® Kintex®-7 410T ■ Logic cells: 406,720 (DSP slices: 1540) ■ Distributed RAM: 5,663 kbit ■ Block RAM: 28,620 kbit
Number of connectors for I/O modules	■ 5	
Device timing	■ 125 MHz	
Internal communication interface	■ IOCNET	
Physical size	■ 238 x 100 x 19 mm (9.4 x 3.9 x 0.7 in) ■ Requires 1 slot plus one additional slot for each I/O module	
Typical power consumption	■ DS2655 7K160: 15 W	■ DS2655 7K410: 30 W

Figure 4.14 - DS2655 technical data

4.5 DS2655M1 – MULTI I/O MODULE

Three I/O modules are available to expanding the I/O channels of the dSpace FPGA base boards. They provide a high number of the digital and analog I/O channels required for application such as electric drives.
In particular, in our project we have used the DS2655M1 Multi I/O module with five analogs in and out channels as well as ten digital channels.
Here are the technical details:

Parameter		Specification
General		■ 5 A/D channels, 5 D/A channels, 10 digital I/O channels ■ 1 x 50-pin Sub-D connector
Analog I/O	Input	■ 5 channels ■ Resolution 14 bit ■ Sampling rate 4 MSPS SAR ■ Input voltage range selectable for each channel: ±5 V or ±30 V
	Output	■ 5 channels ■ Resolution: 14 bit ■ Update rate: 7.8 MSPS ■ Output voltage range: ±10 V
Digital I/O		■ 10 channels, usable as input or output
	Input	■ Maximum input voltage: 15 V ■ Threshold for each channel adjustable from 0 V to +10.5 V
	Output	■ Push-pull drivers ■ One output voltage can be selected for all channels: 3.3 V or 5 V
Physical size		■ 209 x 100 x 19 mm (8.2 x 3.9 x 0.7 in) ■ Mounted on the FPGA base board, requires one additional slot for each I/O module
Typical power consumption		■ 9.6 W

Figure 4.15 - DS2655M1 technical data

5. SOFTWARE MODEL

In this chapter, the methodology and the approach I have used to tackle the project will be explained. In particular, we will examine in detail 4 different models of increasing complexity that have been used to achieve the desired results.

5.1 Methodology/approach to the project

The objective of the thesis was to design and parameterize a Hardware-in-the-loop (HIL) low voltage system (in particular, 48V power supply), consisting of an external control unit developed on a microcontroller, a model of a PMSM motor and an inverter model.
The approach was at to first create a simplified model, consisting of a simple speed control that I implemented on FPGA, in order to test the PMSM motor model implemented on the processor side (electrical + mechanical model).
Then I moved on to a second and more articulated model where the transforms of Clare & Park were introduced in the FPGA part in such a way as to be able to test the latter or mainly to visualize the trend of voltages and currents in each phase.
In the end, the final and complete model of the project was implemented, thus consisting of the PMSM motor, the inverter and the Clarke & Park transforms.
Unfortunately, as we did not have the opportunity to test this model through the use of a real control unit, it was therefore decided to include a model of a controller present in the XSG libraries of the dSpace, in order to be able to view the complete operation of the project.

5.2 Version description

As it has already been anticipated, each single model consists of a part developed in FPGA and another one on a Processor:

Figure 3.1 - Model layout

The same interface was provided to both blocks in order to make the model as tidy and intuitive as possible. It is composed by several blocks:

- **Hardware I/O:** it is a block that manages all the input and output signals from the processor, the FPGA or with any external control unit;
- **Models:** it is a block where there are the models implemented both on the processor and on the FPGA;
- **Signal Routing:** it is an intermediary block that receives the signals from the 'Models block' and sends them in input to the 'Hardware I/O' and vice versa;
- **Parameters:** it is a block where the parameters used to implement the models are present;
- **Control & Multi_scope**: these bloks are implemented in order to display signals and control parameters through the Control Desk.

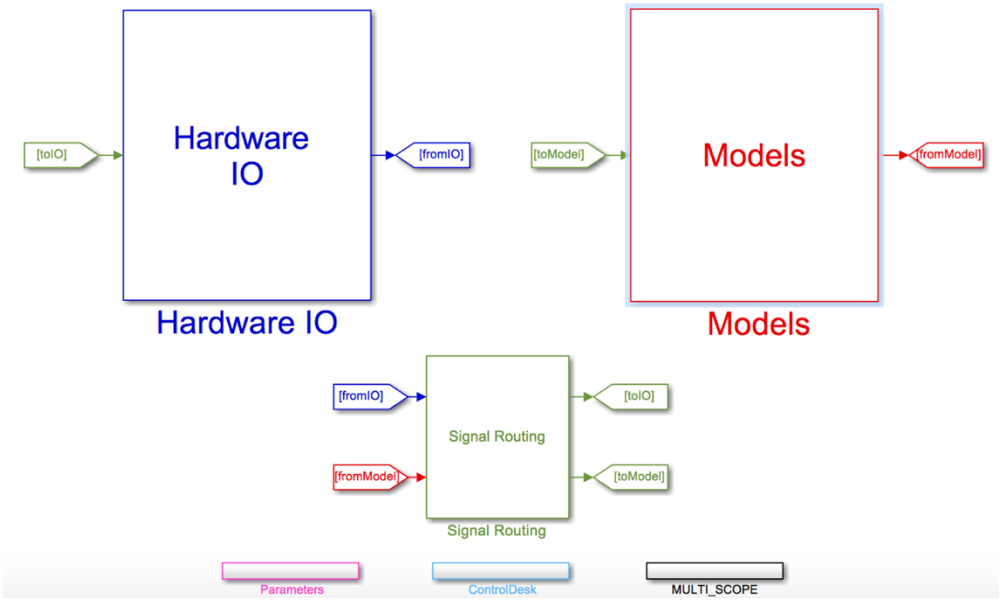


Figure 5.2 - Model interface

5.2.1 Version 1.1

This first version of the model was developed in such a way as to be able to test the model of the permanent magnet motor (PMSM), both for the electric model and for the mechanical model.
Because of the simplicity of this model, we haven't been able to use a real control unit and therefore we decided to implement a simple speed control, that will be herewith explained in details.

In its global and logical setup, the model logically looks like this:

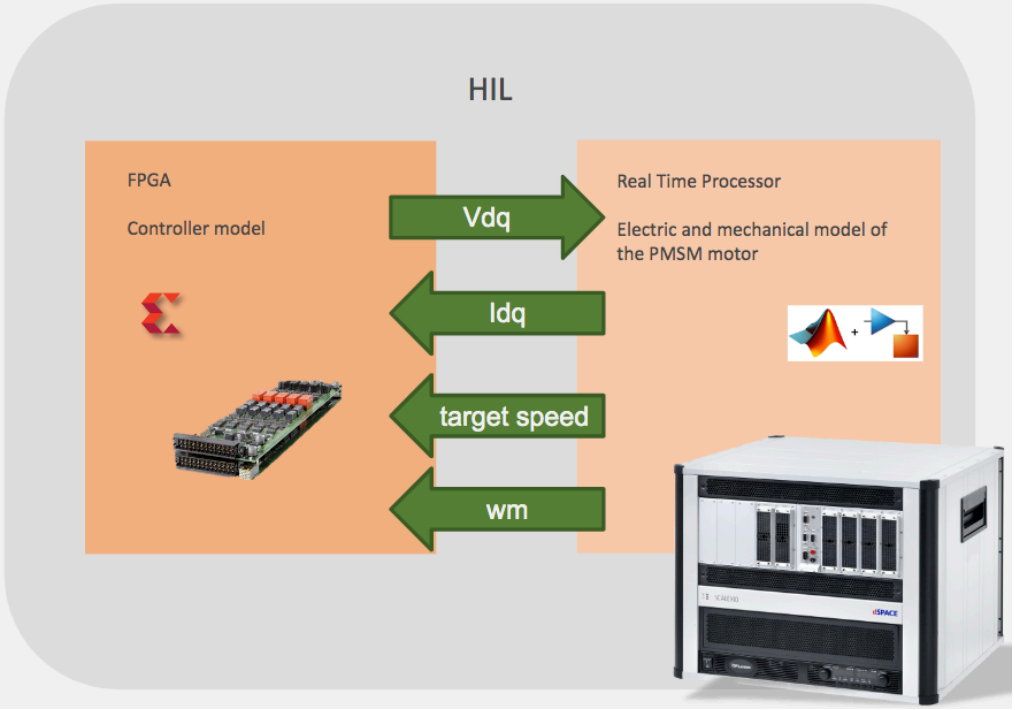


Figure 5.3 - Version 1.1

On the processor side, the version looks like:

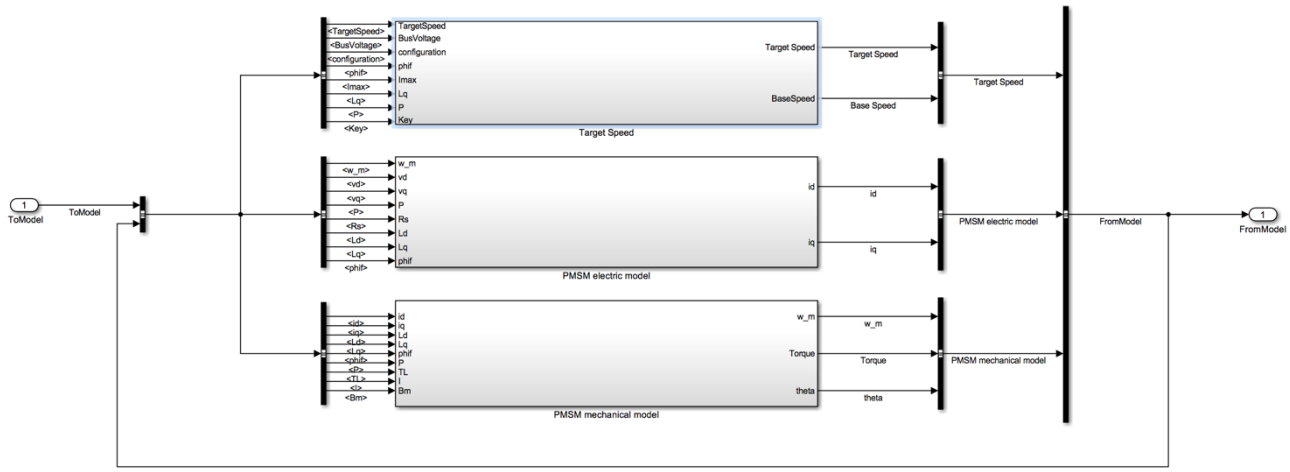


Figure 5.4 – Processor layout of Version 1.1

On FPGA a simple speed control was implemented in this subsystem in order to test the PMSM engine model, so that it followed the target speed provided.

Figure 5.5 - Controller

In details, the developed controller receives in input the currents $i_{d,}$ i_q, the mechanical speed (ω_m) and the target speed (ω_{target}). It also receives proportional and integral coefficients and characteristic parameters of the system. So as to close the model loop, it supplies in output the currents $v_{d,}$ v_q that will enter the electrical model of the motor.

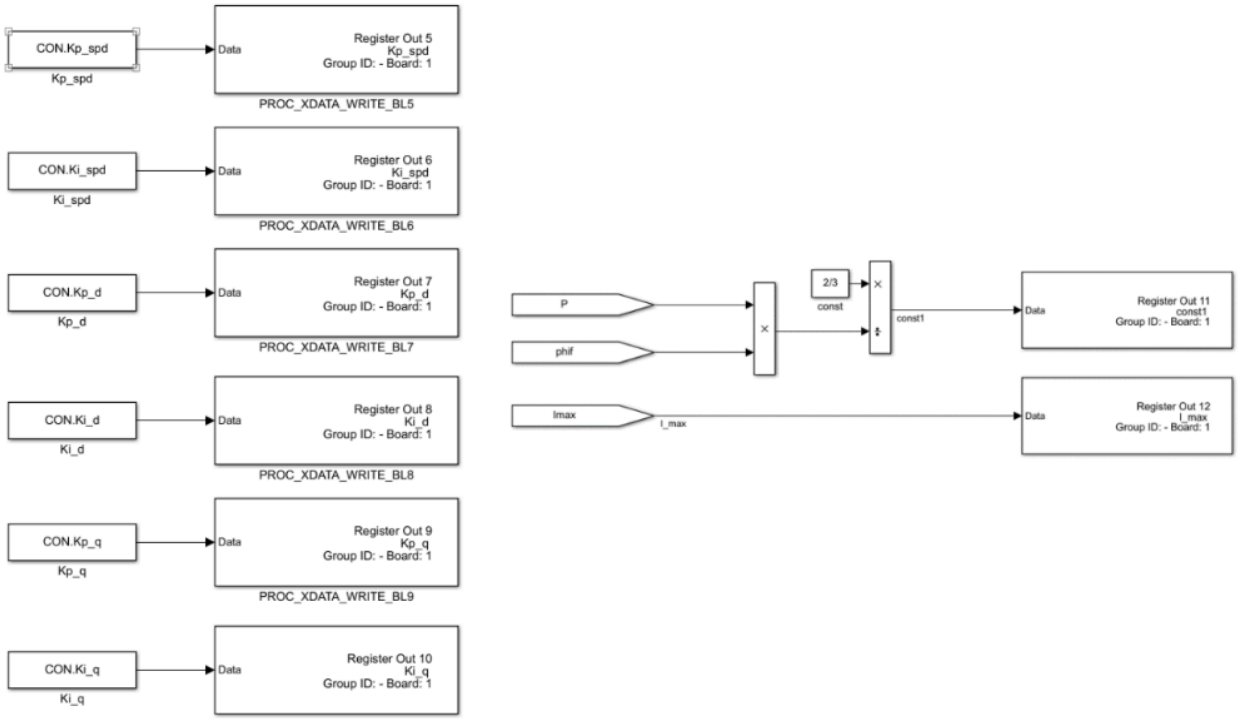


Figure 5.6 – I/O Controller

The implemented controller is therefore a speed control composed of three PIs:

$$H_c(s) = k_p + \frac{k_i}{s}$$

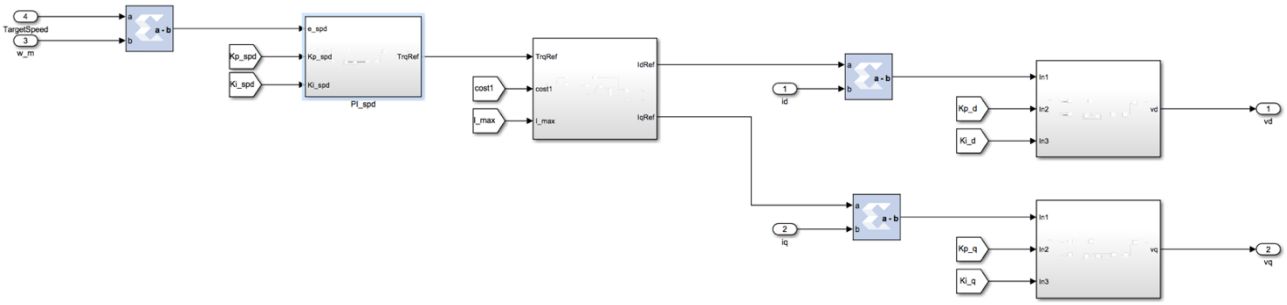


Figure 5.7 – Controller Implementation

Proportional-Integral control (PI) (equal for the 3 PIs present in the sub-system) is totally implemented using the XSG libraries.
The first PI on the left receives the error, that is the difference between the target speed and the motor feedback speed, as an input, and supply as outputs the reference torque T_{ref} which then enters the block that calculates the $i_{dq,ref}$.
On the other hand, the other two PIs receive the difference between the calculated $i_{dq,ref}$ and the i_{dq} from the engine model and the v_{dq} are obtained at the output.

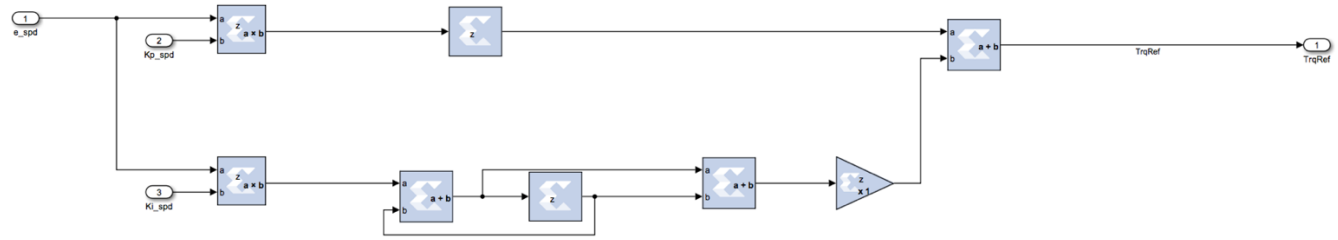


Figure 5.8 – PI Implementation

the main block of the controller consists of:

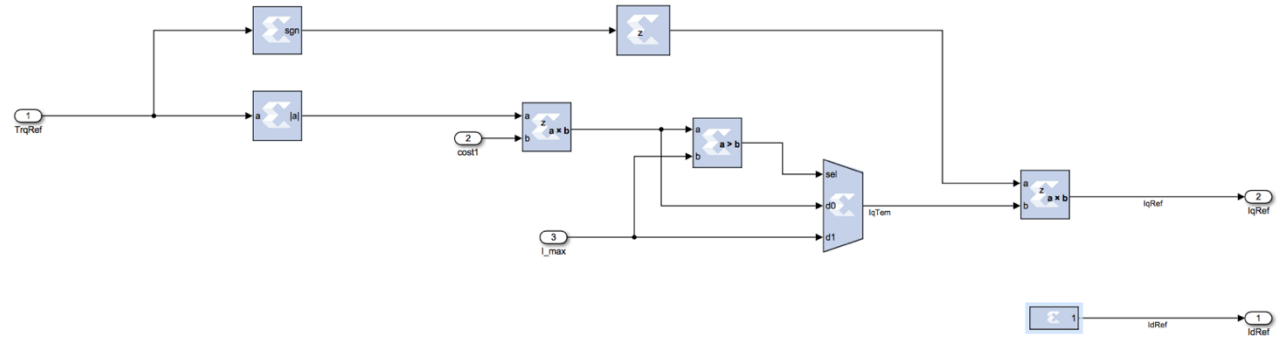


Figure 5.9 - $i_{dq,ref}$ Implementation

where $i_{d,ref} = 0$ in order to avoid that any $i_d \neq 0$ can increase the joule losses in the machine without producing any torque.
$I_{q,ref}$ is instead calculated according to the following reasoning:

$$i_{q,ref} = T_{ref} \cdot i_{q,tmp}$$

where:

$$\begin{cases} if\ |T_{ref}| \cdot const > I_{max} \rightarrow i_{q,tmp} = I_{max} \\ if\ |T_{ref}| \cdot const < I_{max} \rightarrow i_{q,tmp} = |T_{ref}| \cdot const \end{cases} \rightarrow const = \frac{2}{3 * p * \lambda_m}$$

5.2.2 Version 1.2

This second model is to be considered as an upgrade of the model just analyzed. Once the motor model has been deeply and correctly tested, the goal now is to test the Clarke & Park transforms, developed in the FPGA part and to visualize the voltage trend in each phase by means of an oscilloscope.
With the use of a dSpace tool called FPGA Scope, that will be exanimated in detail later, it will also be possible to have a view of the phase currents and voltages in the stationary reference abc.

In its global and logical setup, the model logically looks like this:

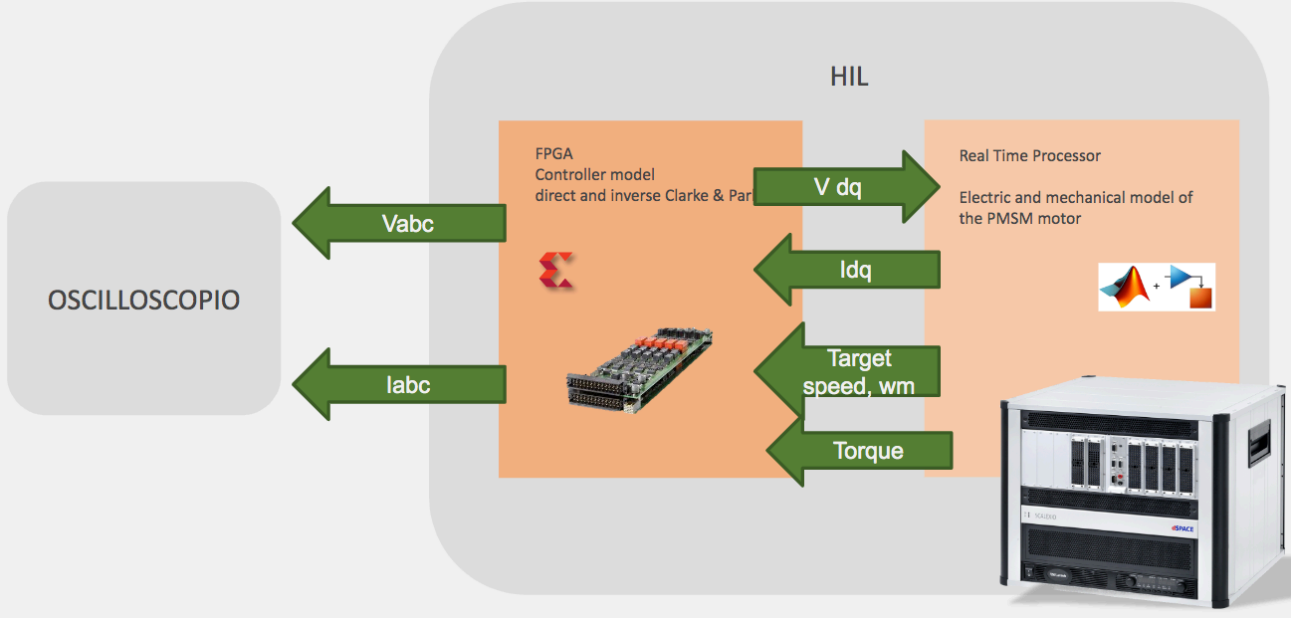


Figure 5.10 – Version 1.2

5.2.3 Version 1.3

Version 1.3 is a transition to the final version 1.4, consisting of the motor model developed on the processor, the inverter model and the Clark & Park transforms developed on the FPGA. In this version the controller running on the FPGA is deleted. In addition, I/O channels are added. In particular 3 'digital in' which allows us to read the PWM from the controller and 3 'analog out' in order to feed back the three phase currents in the three-phase reference.
With the addition of a model to manage the CAN communication with the real control unit (to communicate the desired target speed and other useful information), at the moment not yet implemented, it will be possible to operate this version with a real control unit.

In its global and logical setup, the model logically looks like this:

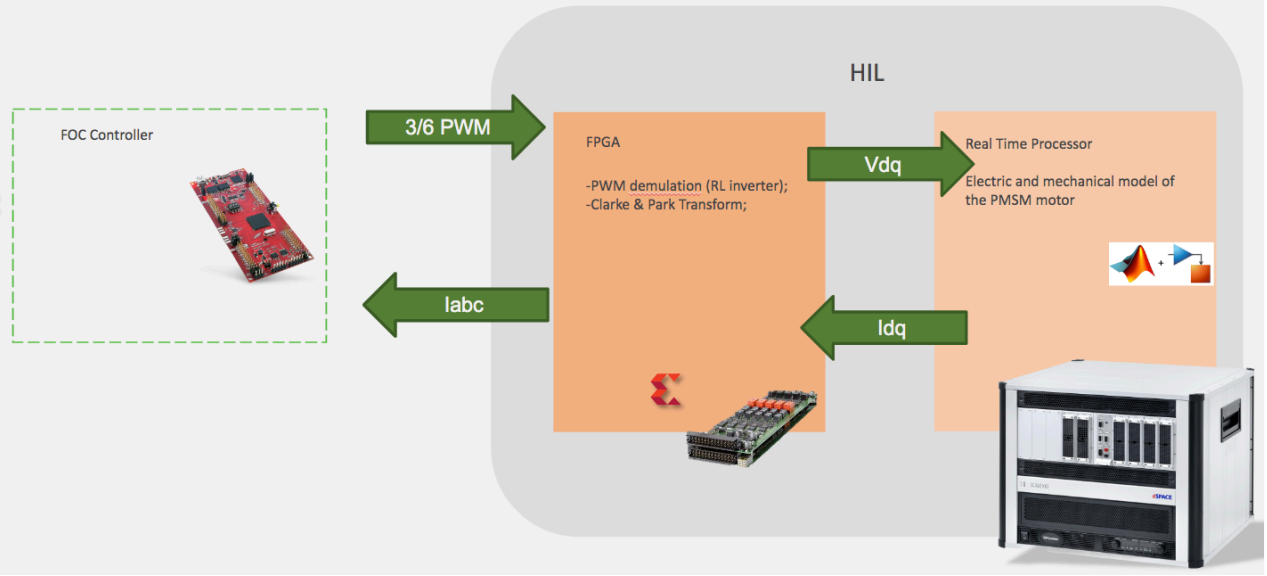


Figure 5.11 – Version 1.3

5.2.4 Version 1.4

As we could not use a real control unit, this has been replaced by a controller model developed using the library dSpace, a model that has been already extensively analyzed in the previous chapters.
Later, there will be a detailed explanation of the version 1.4, final version of the PMSM model.

In its global and logical setup, the model logically looks like this:

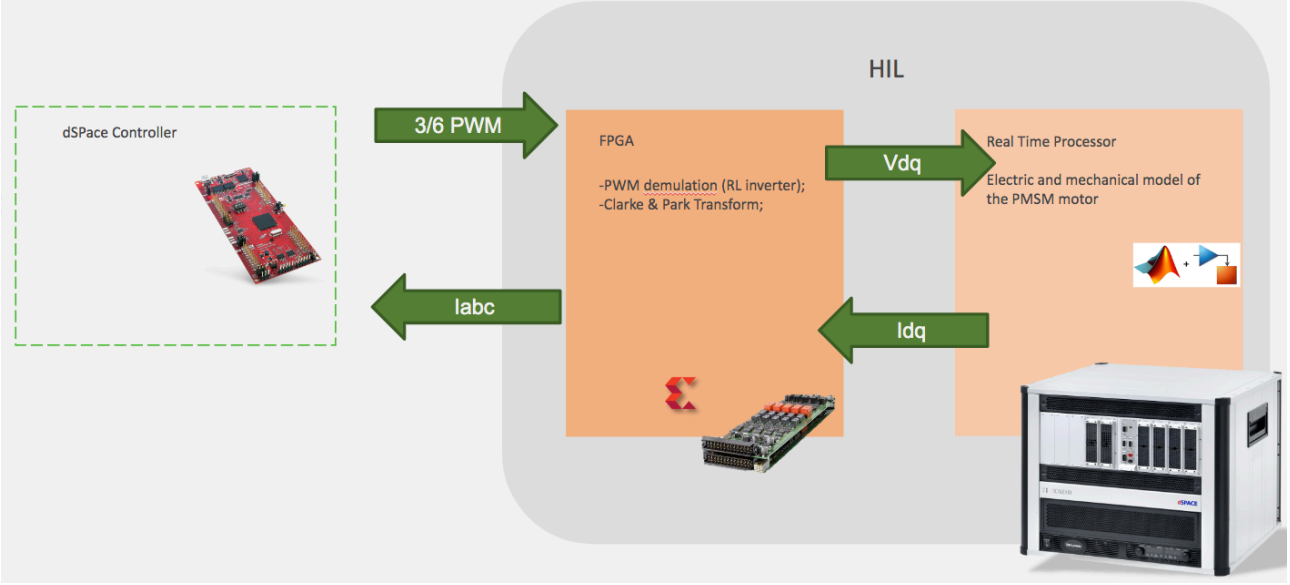


Figure 5.12 – Version 1.4

5.2.4.1 Processor

In the model that runs on the processor there is the calculation of the target speed, the model of the electric motor divided into electrical and mechanical equations:

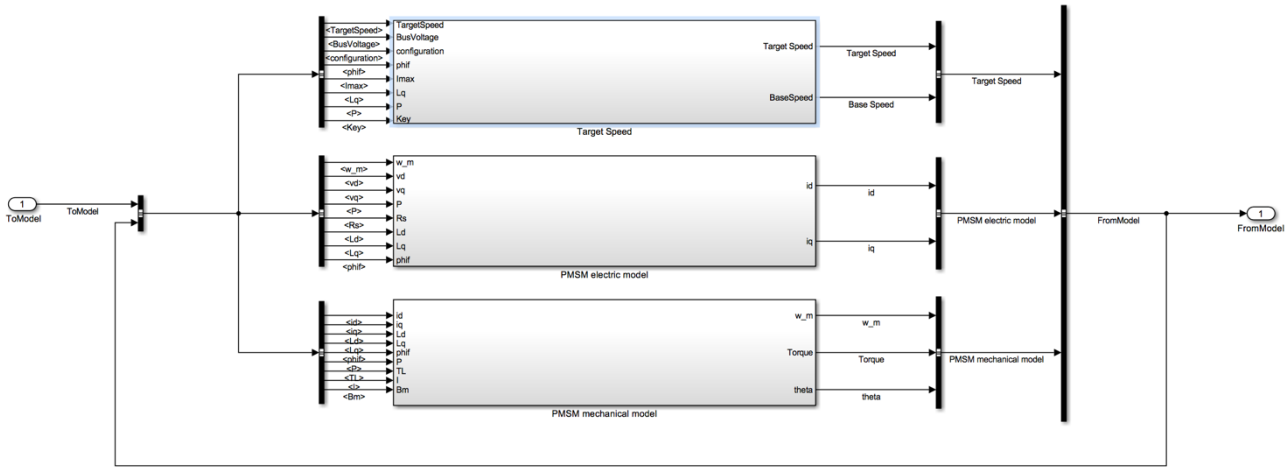


Figure 5.13 – Processor of Version 1.4

- ‘Target Speed’:

The Target Speed subsystem receives as input:

- Target Speed: indicates the desired mechanical speed and is entered in real-time by the user on the Control Desk;
- Key: signal used to activate or deactivate the speed request;
- Configuration: indicates the configuration of the electric motor, if it is a star or triangle, and it can be modified in real-time. It is used for the calculation of the base Speed;
- Various parameters of the motor system for the calculation of the base Speed.

The idea of the 'target speed' block is therefore to ensure that a speed higher than the basic motor speed will not be permitted as the controller does not consider the flux-weakening.

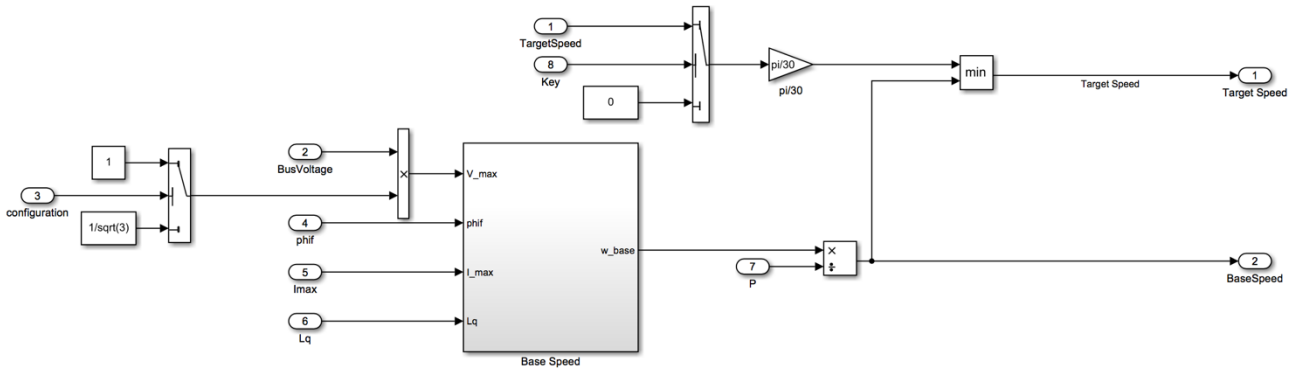


Figure 5.14 – Target Speed block

The base speed has been calculated on the basis of the parameterization we have made and in particular it takes into account the configuration of the motor: star or delta.

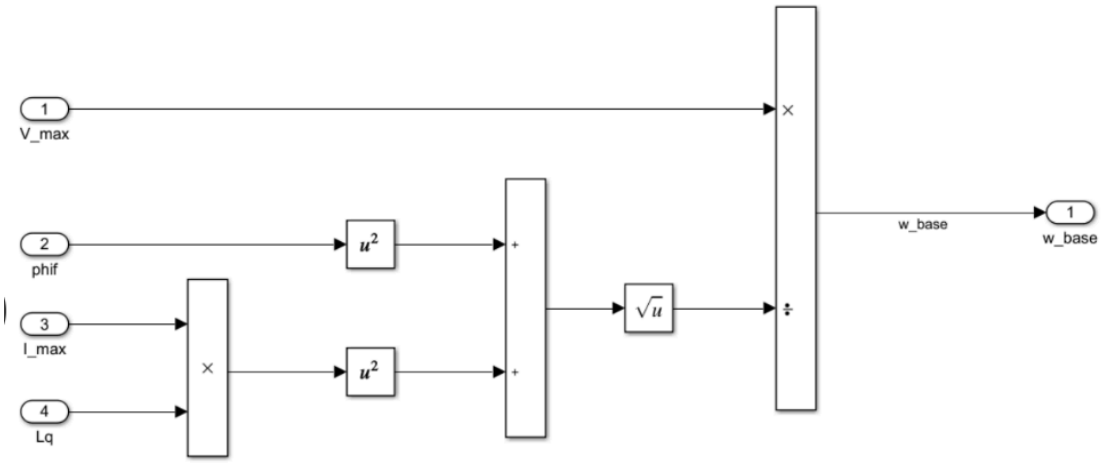


Figure 5.15 – Base Speed block

This simple block was implemented by following the equation for calculating the base speed:

$$\omega_b = \frac{V_N}{\sqrt{L^2 I_N^2 + \lambda_m^2}}$$

The Target Speed subsystem outputs the desired saturated mechanical base speed and the calculated base speed.

- 'PMSM Electric model':

The PMSM Electric Model subsystem receives as input:

- Mechanical speed of the previous step;
- The tensions in the stationary reference dq coming from the model of the inverter present in the FPGA;
- PMSM parameters: stator resistance (R_s), inductances in the reference dq (L_d and L_q), pole pairs (P) and magnetic flux of the magnets permanent (λ_m).

Concerning the electrical model, I have considered the following equation of the Permanent magnet synchronous motor:

$$\begin{cases} v_d = R_s i_d + \dfrac{d\lambda_d}{dt} - \omega \cdot \lambda_q \\ v_q = R_s i_q + \dfrac{d\lambda_q}{dt} + \omega \cdot \lambda_d \end{cases}$$

then substituting $\lambda_d = L_d i_d + \lambda_m$ and $\lambda_q = L_q i_q$ in the last equation and solving as a function of i_d and i_q we obtain that:

$$\begin{cases} \dfrac{di_d}{dt} = \dfrac{v_d - R_s i_d + \omega \cdot L_q i_q}{L_d} \\ \dfrac{di_q}{dt} = \dfrac{v_d - R_s i_q + \omega \cdot L_d i_d - \omega \cdot \lambda_m}{L_q} \end{cases}$$

Finally, by integrating this last equation, we obtain the currents i_{dq} which will then be sent to the FPGA input and used by the controller.

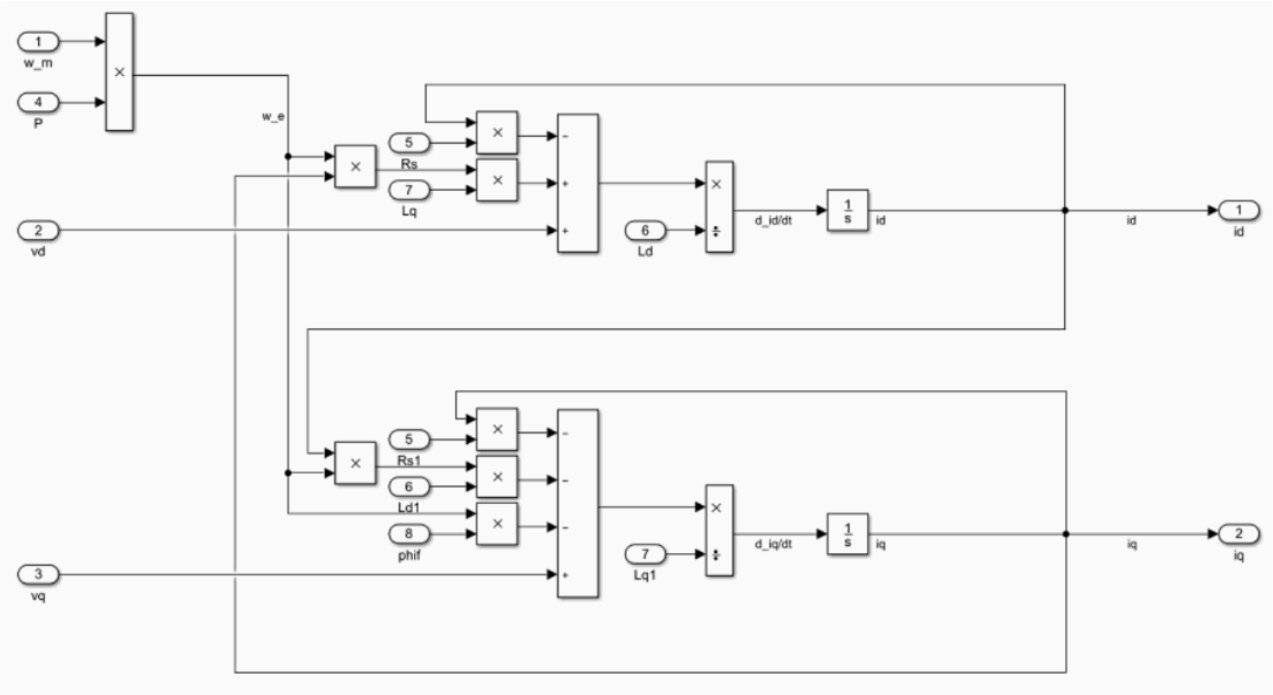


Figure 5.16 – Electric model block

- 'PMSM Mechanical model':

The PMSM Mechanical Model subsystem receives these inputs:

- the currents in the dq stationary reference from the PMSM Electric Model subsystem;
- the load torque T_L;
- engine and inverter parameters.

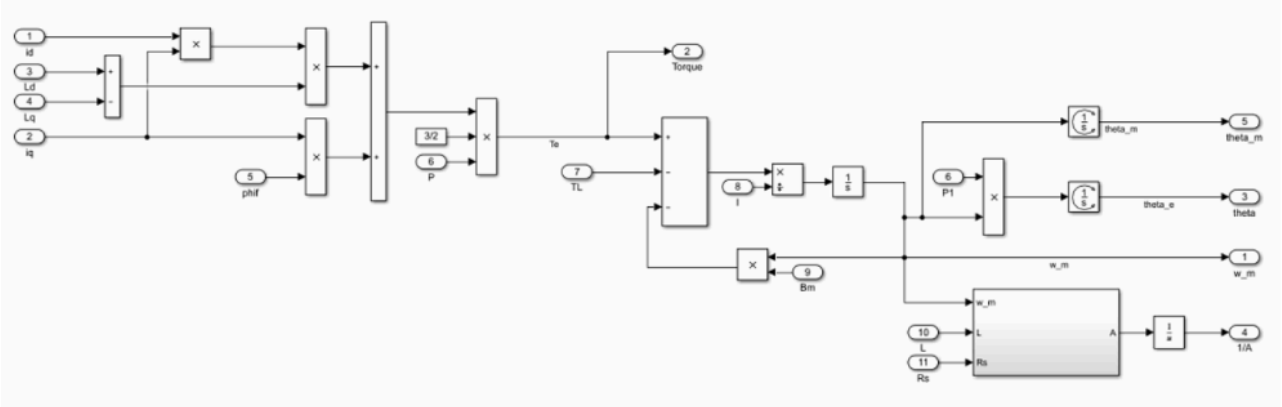


Figure 5.17 – Mechanic model block

At first, the motor torque was calculated using the following formula obtained from the PMSM motor model:

$$T_e = \frac{3}{2} p\left[\lambda_m i_q + (L_d - L_q) i_d i_q\right]$$

The mechanical speed and the electrical angle were then calculated respectively.

$$\frac{d\omega_m}{dt} = T_e - T_L - \omega_m \cdot \beta_m \quad , \quad \theta_e = p \cdot \int \omega_m \, dt = p \cdot \theta_m$$

The processor also has an interface between the processor and the FPGA for using the FPGA Scope in the MULTI_SCOPE subsystem.

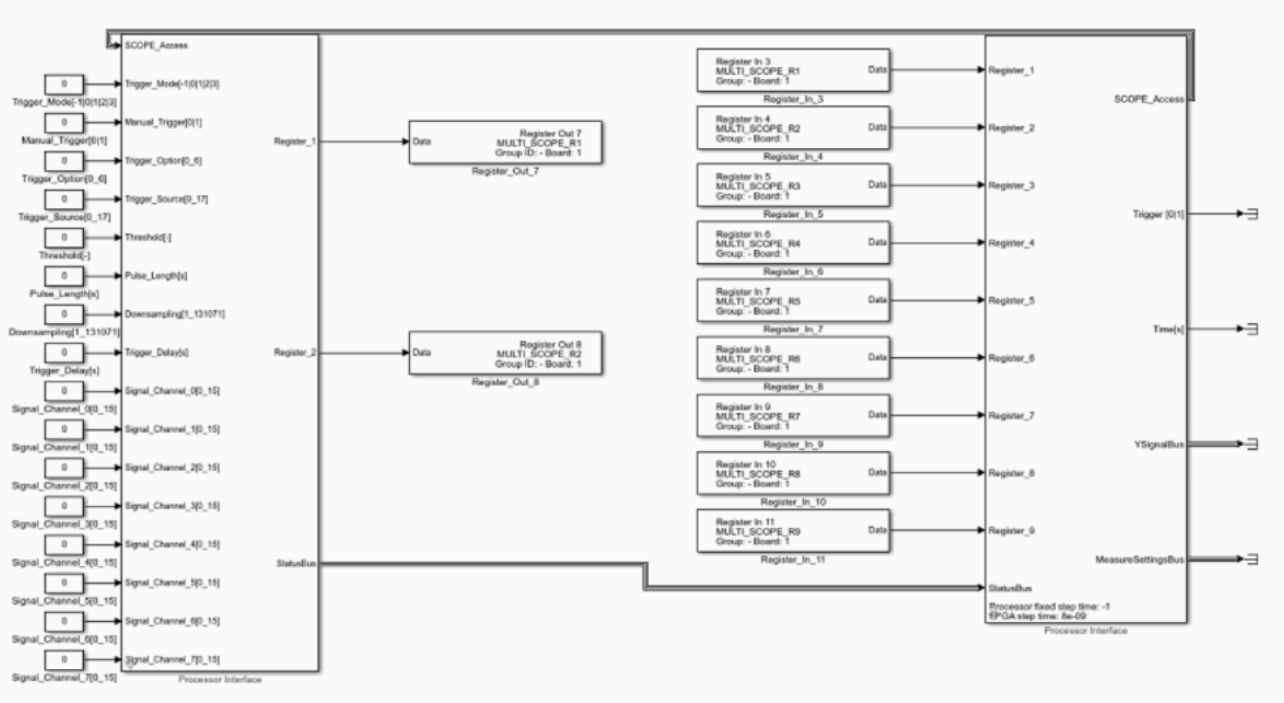


Figure 5.18 – MULTI_SCOPE block

5.4.2.2 FPGA

In the model that runs on the FPGA, the inverter model and the transforms of Clarke and Park direct and reverse are implemented.

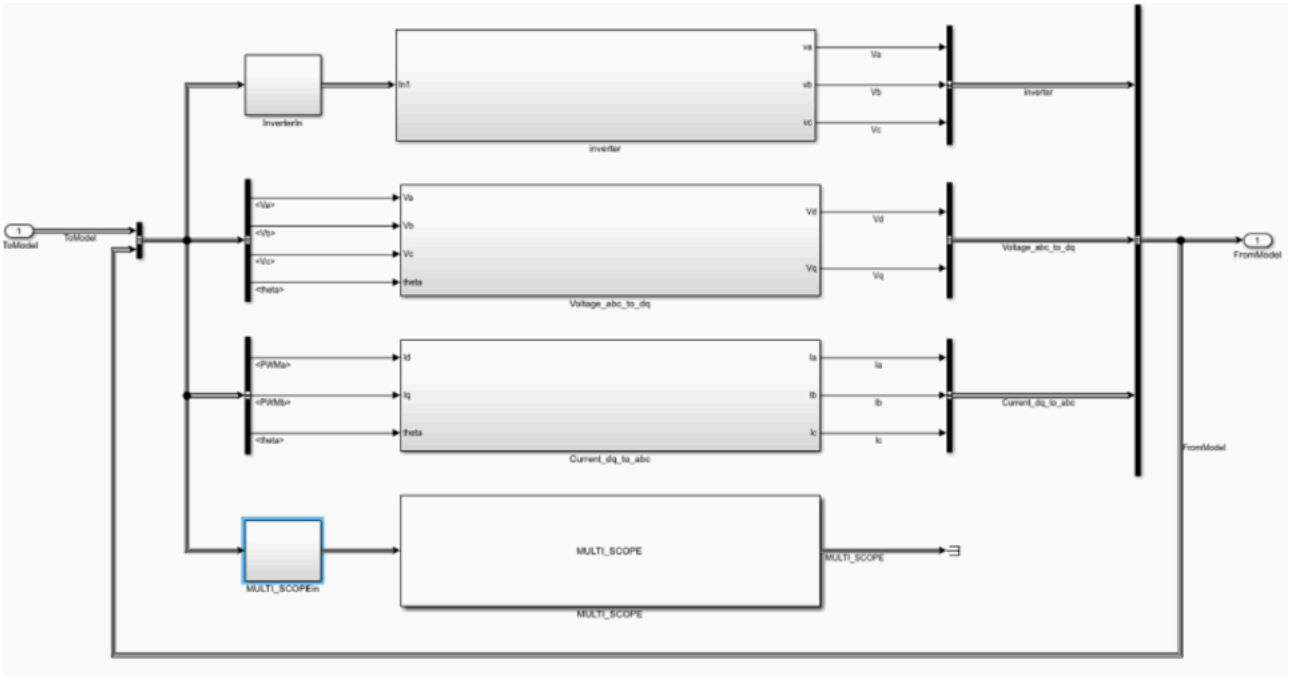


Figure 5.19 – FPGA of Version 1.4

- Inverter:

The 'Inverter' subsystem receives input:

- the 3 PWM from the controller to implement the inverter switches;
- the supply voltage;
- other inverter related parameters.

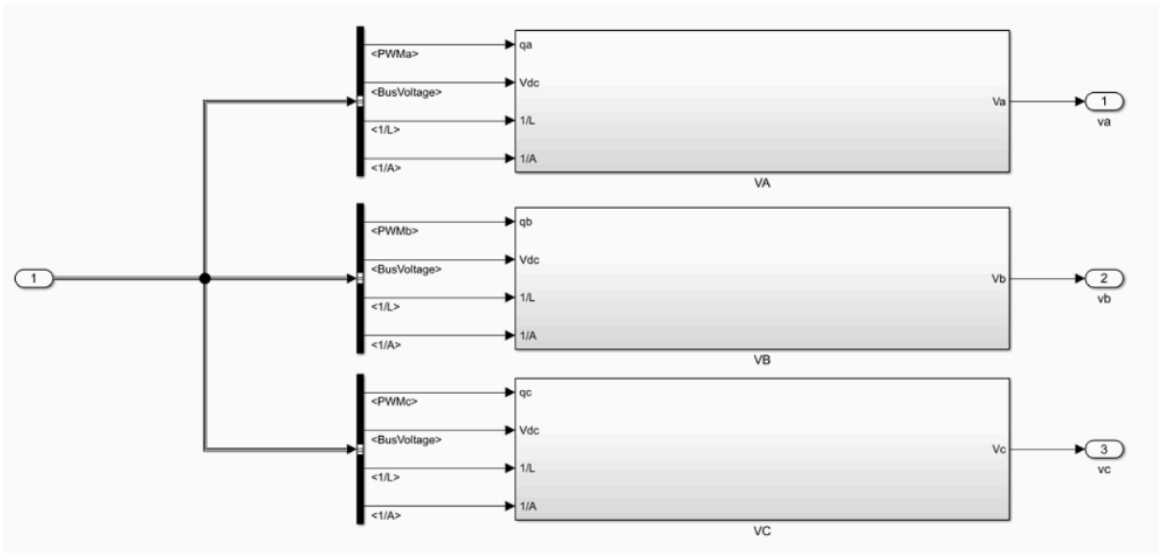


Figure 5.20 – Inverter block

This subsystem is divided into three identical blocks that calculate in a similar way the three different voltages in each phase corresponding to the PWM.
The inverter consists of a simple RL circuit where the level of each of the three PWM is compared to a threshold:

$$\begin{cases} if\ q_{A,B,C} \geq 0.5 \ \rightarrow \ u(t) = V_{dc} \\ if\ q_{A,B,C} < 0.5 \ \rightarrow \ u(t) = -V_{dc} \end{cases}$$

The presented RL circuit can be modelled through the following block diagram:

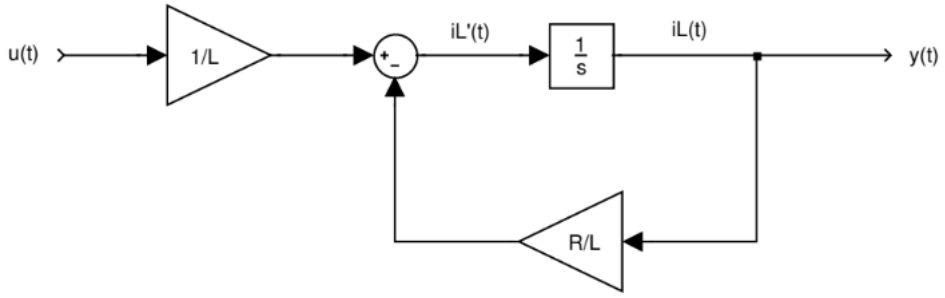


Figure 5.21 – RL circuit block diagram

the output signal is therefore a sinusoid of smaller amplitude and is attenuated by a magnitude equal to:

$$A = \frac{\sqrt{L^2\omega^2 + R^2}}{L^2\omega^2 + R^2}$$

to reach the desired amplitude, the output signal has been multiplied by a factor equal to $^1/_A$.

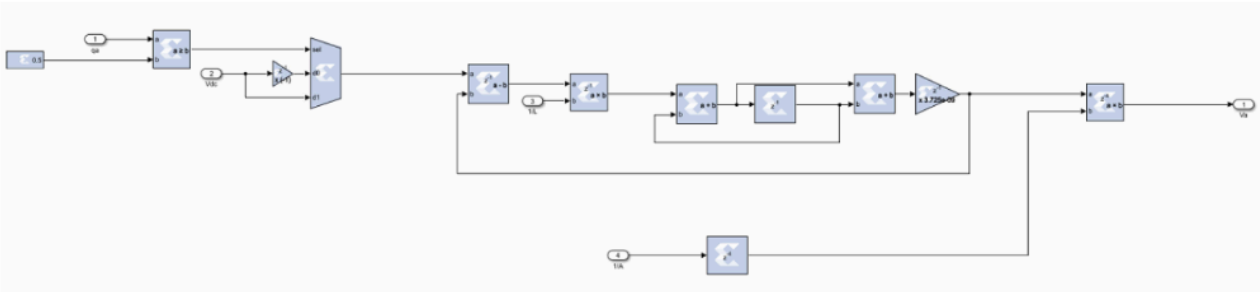


Figure 5.22 – Vabc block

On the way out we have therefore v_{abc} that will be then used in the transforms of Clarke and Park to change reference, from abc to dq, and to enter the model of the engine.

- Clarke & Park transform:

The 'Voltage_abc_to_dq' subsystem is responsible for transforming the phase voltages, in the stationary reference abc, in the rotating reference dq, v_d and v_q.

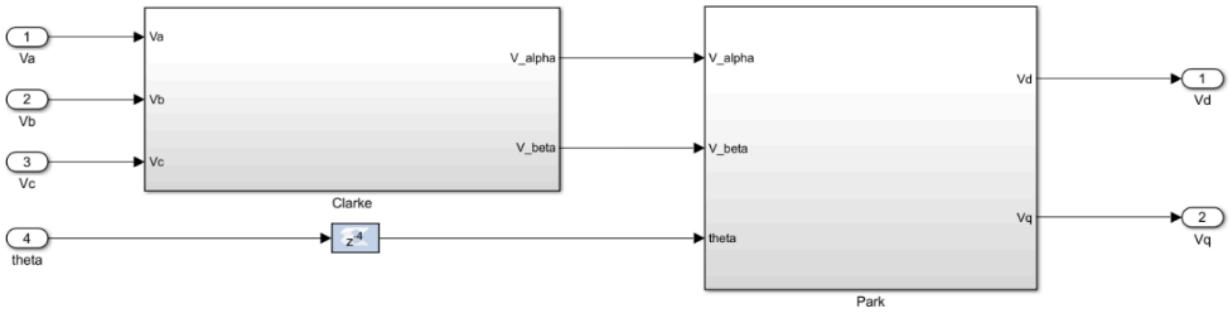


Figure 5.23 – Clarke & Park transform

This block is divided into two subsystems:

- Clarke: in order to implement the Clarke transform, to switch from a three-phase system (abc) to an orthogonal stationary frame $(\alpha\beta)$, the following equation is used:

$$\begin{bmatrix} x_\alpha \\ x_\beta \\ x_o \end{bmatrix} = [T] \cdot \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \quad \leftrightarrow \quad [T] = \frac{2}{3} \cdot \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

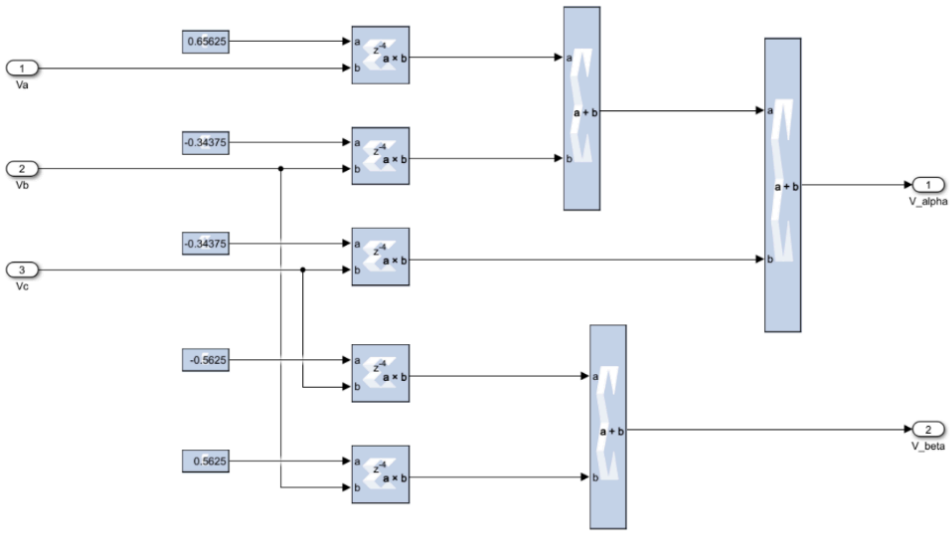


Figure 5.24 – Clarke transform

- Park: in order to implement the Park transform, to switch from a orthogonal stationary frame $(\alpha\beta)$, to an orthogonal rotating frame (dq), the following equation is used:

$$\begin{bmatrix} x_d \\ x_q \end{bmatrix} = \begin{bmatrix} cos\vartheta & sin\vartheta \\ -sin\vartheta & cos\vartheta \end{bmatrix} \cdot \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = [A(\vartheta)] \cdot \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix}$$

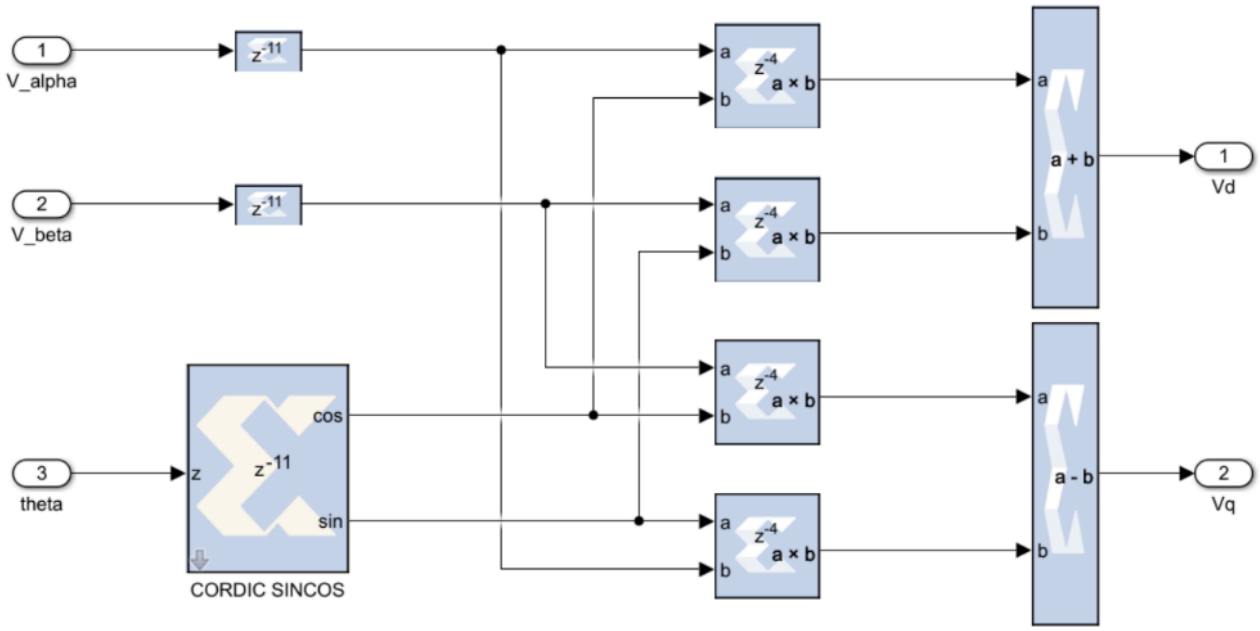


Figure 5.25 – Park transform

- Clarke & Park inverse transform:

The subsystem 'Current_dq_to_abc' has the task of transforming the phase currents, in the rotating reference dq, in the stationary reference abc.

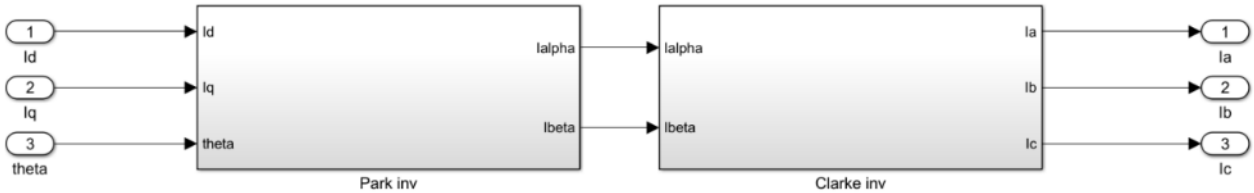


Figure 5.26 – Inverse Clarke & Park transform

This block is divided into two subsystems:

- Park inverse: in order to implement the inverse Park transform, to switch from an orthogonal rotating frame (dq) to an orthogonal stationary frame$(\alpha\beta)$ the following equation is used:

$$\begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = \begin{bmatrix} cos\vartheta & -sin\vartheta \\ sin\vartheta & cos\vartheta \end{bmatrix} \cdot \begin{bmatrix} x_d \\ x_q \end{bmatrix} = [A(-\vartheta)] \cdot \begin{bmatrix} x_d \\ x_q \end{bmatrix}$$

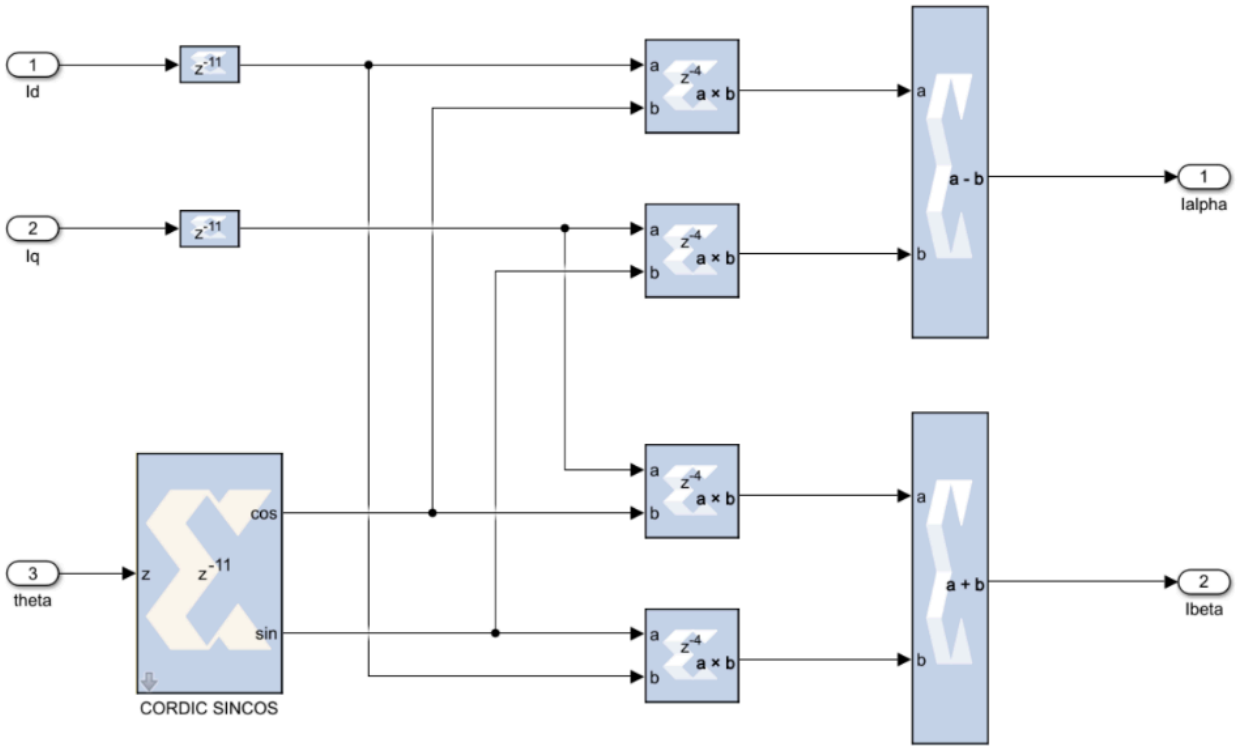


Figure 5.27 – Inverse Park transform

- Clarke inverse: in order to implement the inverse Clarke transform, to switch from an orthogonal stationary frame $(\alpha\beta)$ to a three-phase system (abc),the following equation is used:

$$\begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} = [T]^{-1} \cdot \begin{bmatrix} x_\alpha \\ x_\beta \\ x_0 \end{bmatrix} \quad \leftrightarrow \quad [T]^{-1} = \frac{2}{3} \cdot \begin{bmatrix} 1 & 0 & 1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & 1 \end{bmatrix}$$

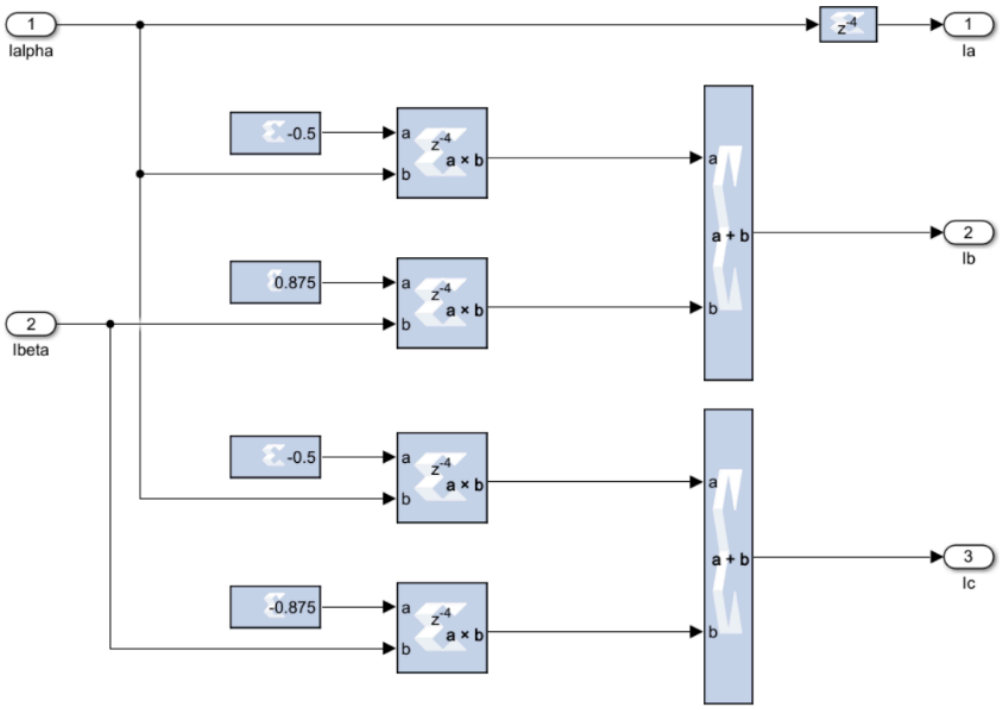


Figure 5.28 – Inverse Clarke transform

- Input/Output:

The FPGA receives as input the 3/6 PWM generated by the controller through the use of Digital In channels.

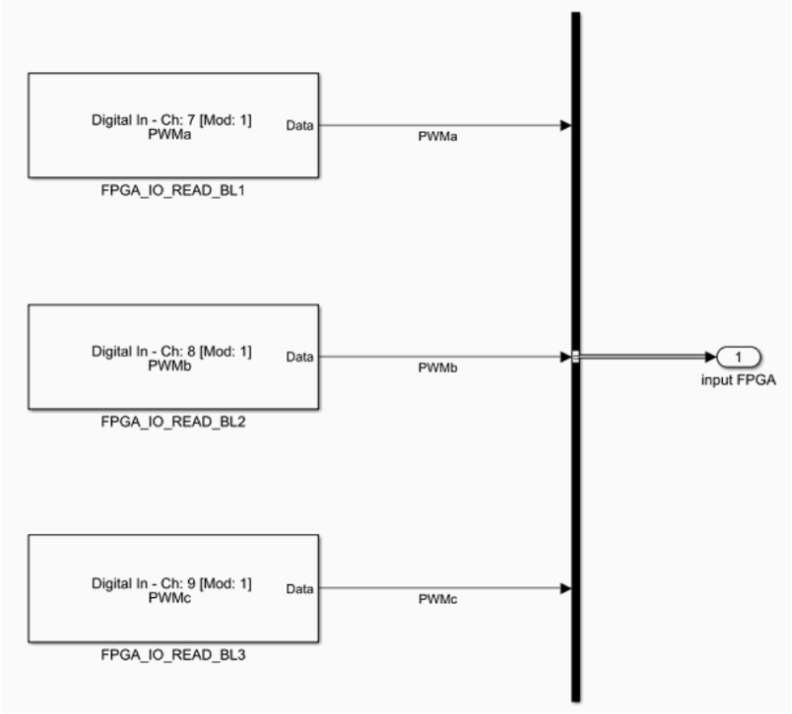


Figure 5.29 – Input of FPGA

The inverter model in FPGA calculates the v_a, v_b and v_c phase voltages. By using the use of the 'Voltage_abc_to_dq' model, they are converted into v_d and v_d, which are then passed to the processor, through the use of communication blocks between FPGA and processor.

The processor with the PMSM electric model and PMSM mechanical model calculates i_d and i_q currents, mechanical speed, mechanical position and torque.

The currents in the reference dq are passed to the FPGA that by using the 'Current_dq_to_abc' model transforms them into i_a, i_b and i_c phase currents.

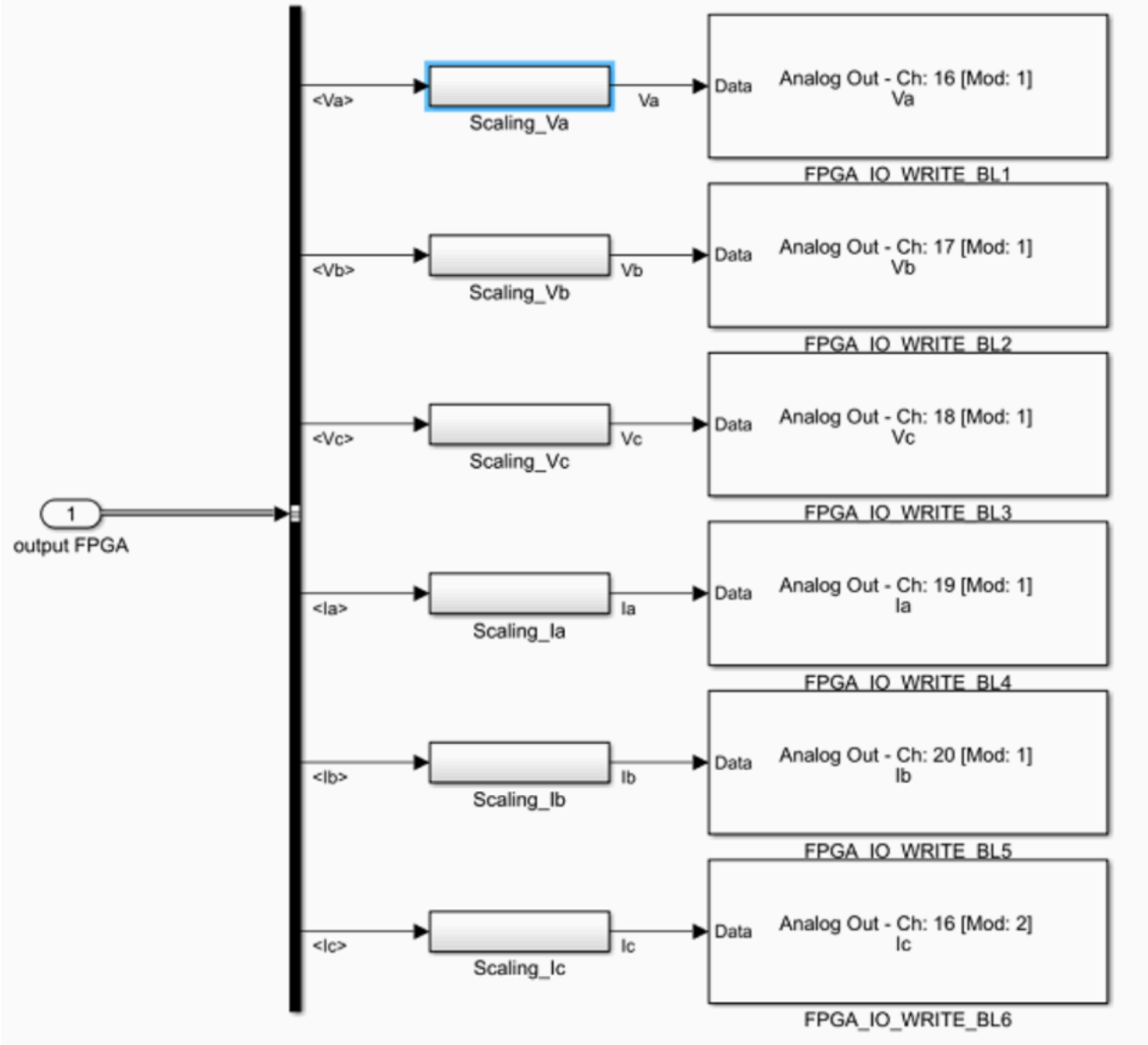


Figure 5.30 – Output FPGA

5.3 FPGA Scope

Since version 1.2, a FPGA scope has been implemented in the model, using dSpace's XSG Electric libraries.The FPGA Scope has the ability to see signals that evolve quickly in the FPGA, with a calculation step of 8 ns, taking advantage of a data buffer towards the processor.
The MULTI_SCOPE block allows the automatic generation of the interface processor-FPGA, creating two temporary files that must be integrated into the model.

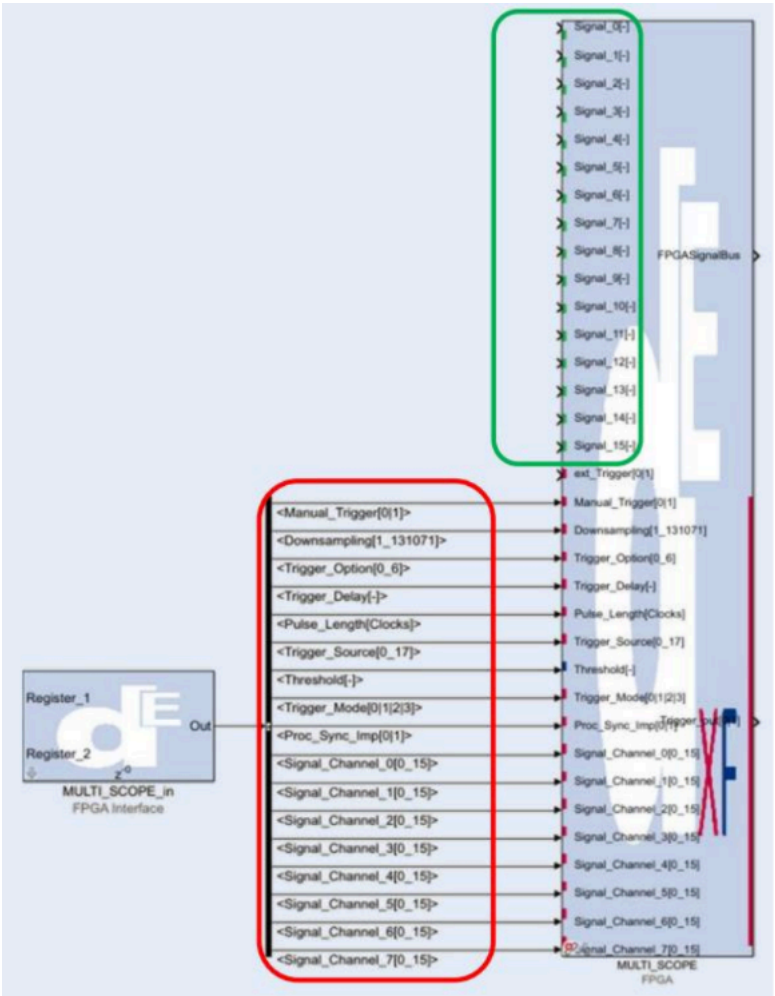


Figure 5.31 – FPGA interface of MULTI_SCOPE

In the Multiscope block in the FPGA part you must connect the signals you want to display in the scope on ControlDesk to the Signal ports [0_15]. The signals below are for checking the scope.
You can connect 16 signals and 8 of them can be seen simultaneously on ControlDesk.

On the processor side you can control the triggering and capture signals.
Trigger modes are similar to a real oscilloscope: 0 = stop, 1 = run, 2 = signal sequence, 3 = video:

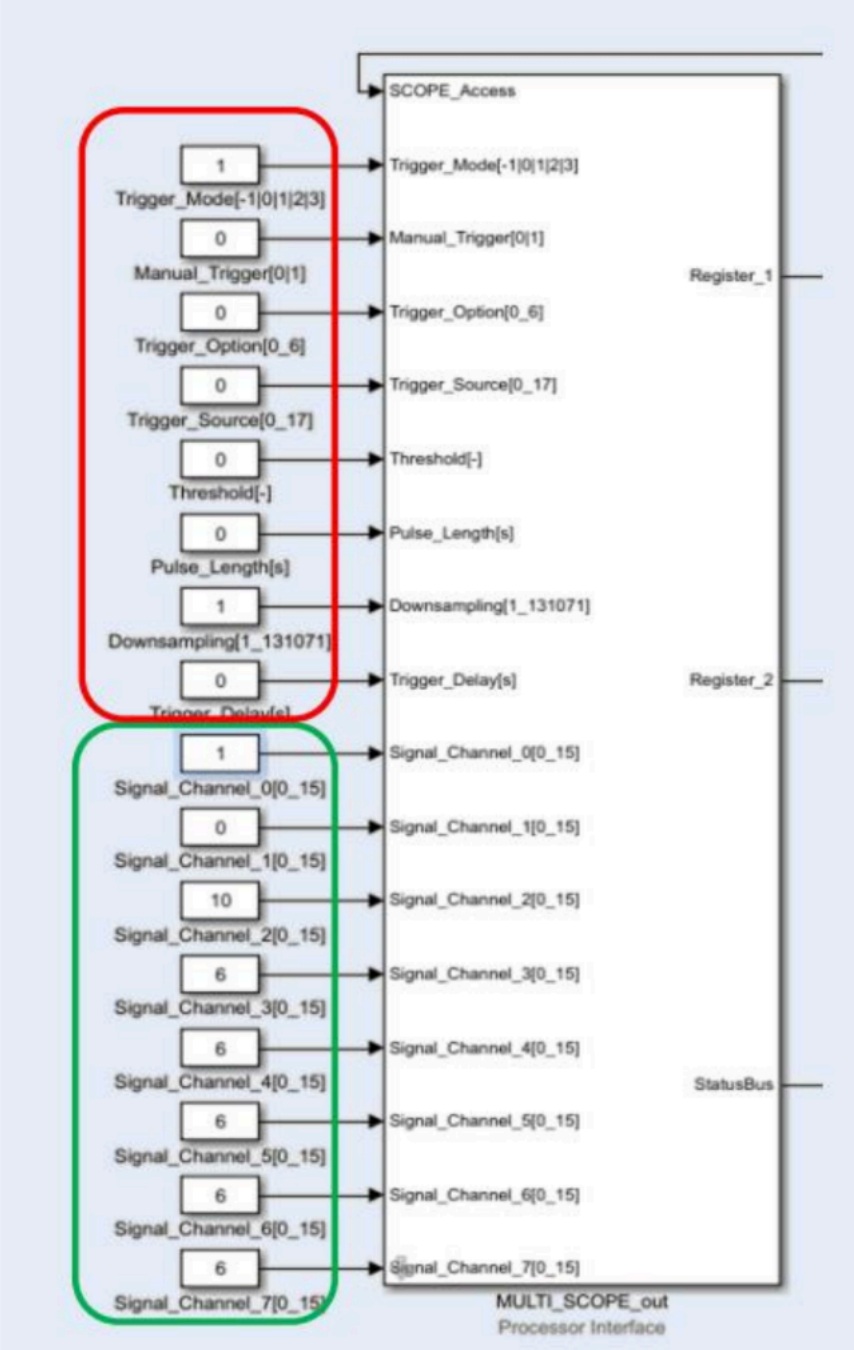


Figure 5.32 – Processor interface of MULTI_SCOPE

To use Multiscope, the XSG Utils library must be imported into ControlDesk. Dragging any variable in the Multiscope will start the automatic configuration and at the same it will generate the scope and the settings in order to control the oscilloscope.

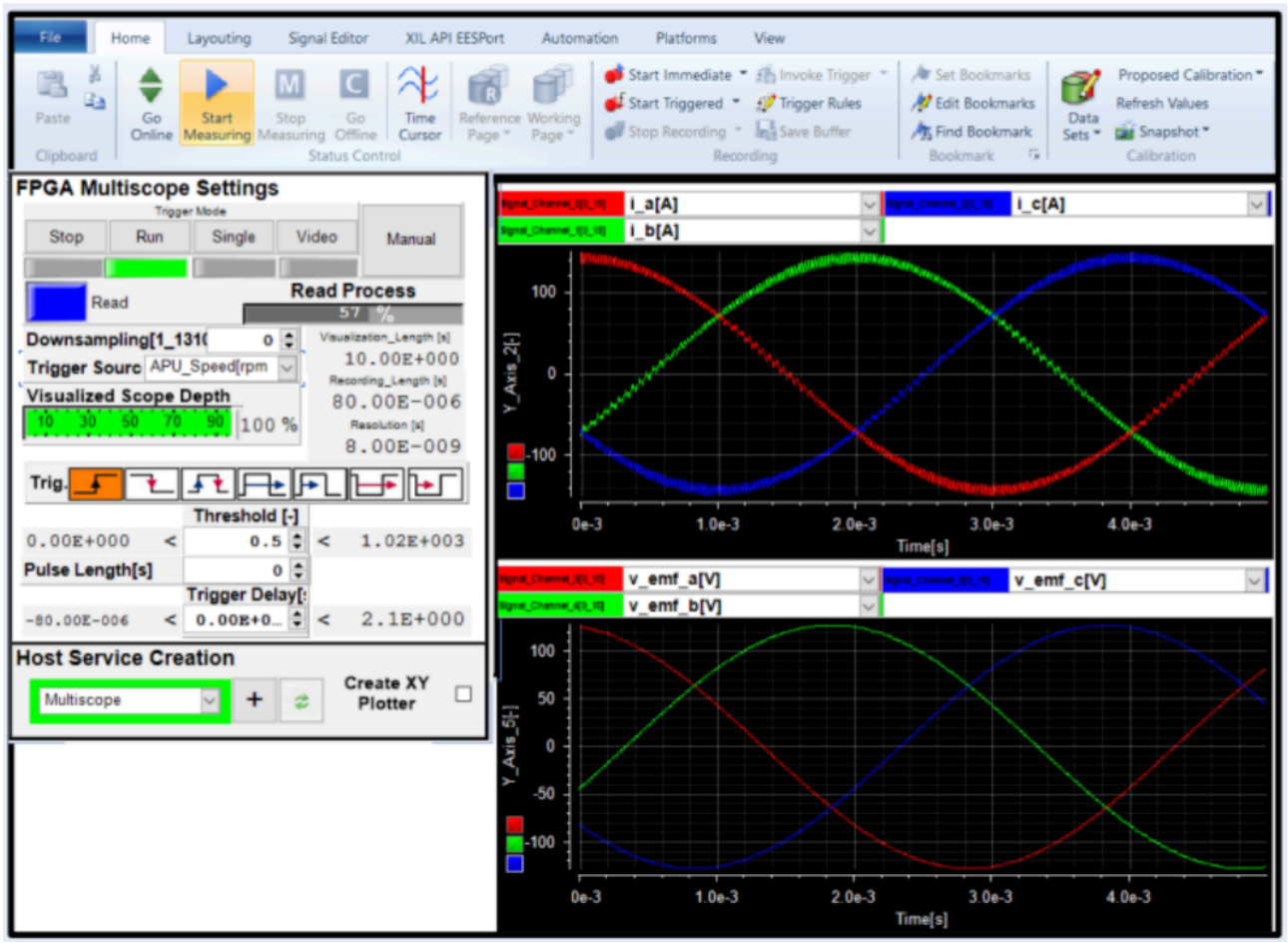


Figure 5.33 – FPGA scope interface

5.4 Configuration

After developing the model that will run on FPGA, there are two important steps in order to load the model on the FPGA and test it through the use of the simulator:

- Timing analysis: it is a test that allows you to check if each block and operation of the implemented model are performed within the set latencies or if more time is required.

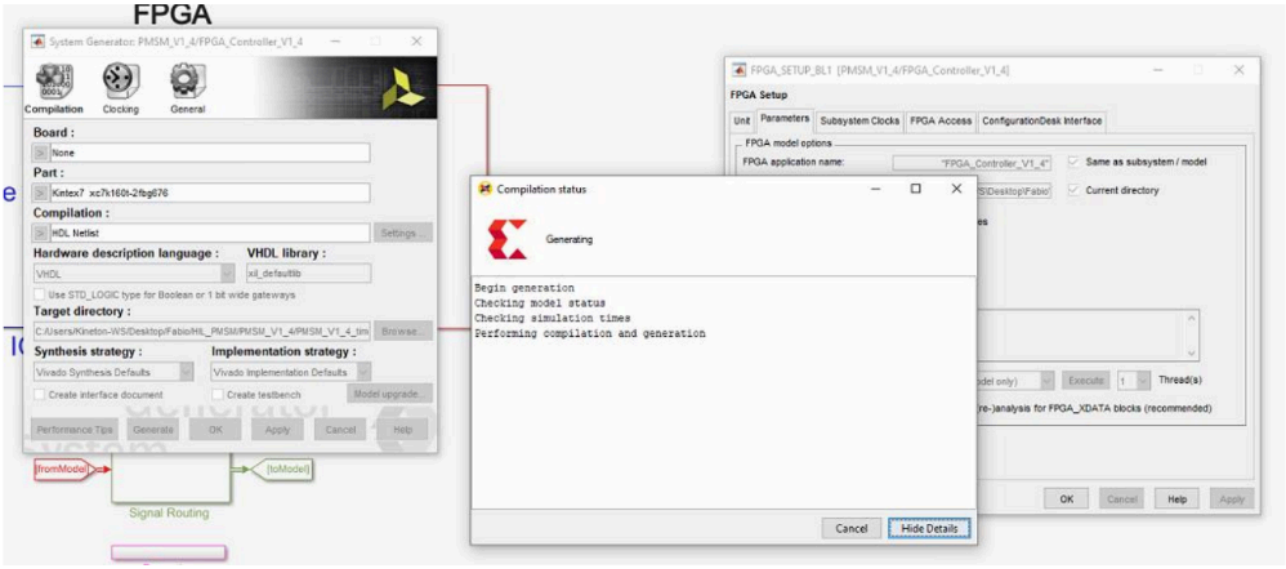


Figure 5.34 – Timing analysis

- Build: it is a function that allows you to translate the language implemented on Simulink into HDL code. Once the process is finished, a folder with an .ini file inside will be created so that you can configure the FPGA part on Configuration desk.

```
WORK   DIRECTORY: C:\Users\Kineton\Desktop\Andreas\ControllerInverter\inverter_rtiFPGA\FPGA_5744FAAAEA2D36
BUILD  DIRECTORY: C:\Users\Kineton\Desktop\Andreas\ControllerInverter\inverter_rtiFPGA\FPGA_5744FAAAEA2D36
RESULT FILE:      C:\Users\Kineton\Desktop\Andreas\ControllerInverter\inverter_rtiFPGA/INI/FPGA_5744FAAAEA2D36.ini

                                          Type    Used   Available   Utilization [%]
   Configurable Logic Block Slices (LUTs, Flip-Flops)    6509       25350           25.68
                Configurable Logic Block Slice LUTs   11744      101400           11.58
          Configurable Logic Block Slice Flip-Flops   17827      202800            8.79
                           Block RAM Blocks 36 Kb       8         325            2.46
                           Block RAM Blocks 18 Kb      12         650            1.85
                                       DSP Slices      24         600            4.00

FPGA Build Done
Elapsed time is 00:25:54.
```

Figure 5.35 – Build

Once you have completed these two operations and then the code of the FPGA has been generated, the next step is the configuration of the model through Configuration Desk.
Through the FPGA setup block you can automatically generate a Configuration Desk Project with all connections for communication between FPGA and processor.

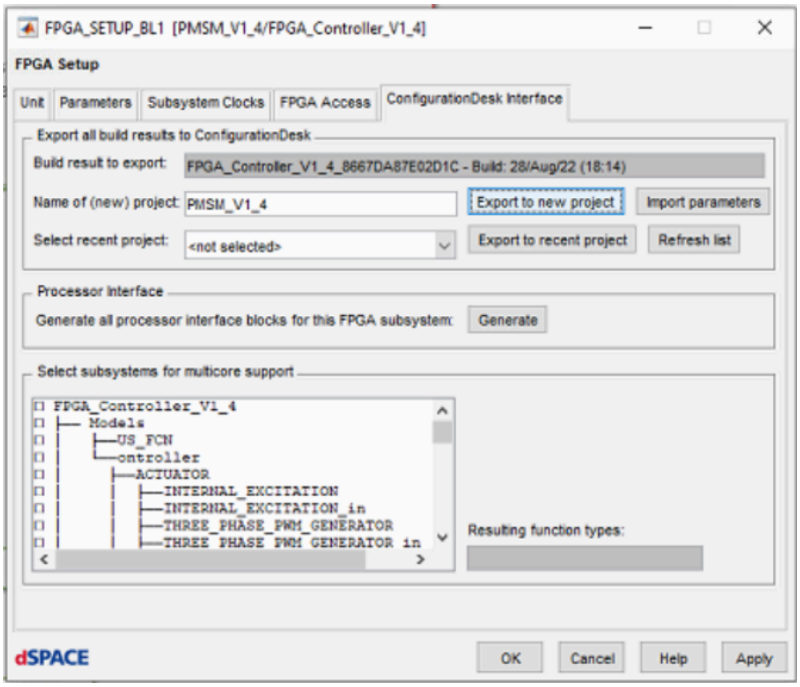


Figure 5.36 – FPGA Setup block

To create this project, a specific block located in the dSpace libraries called Model Separation Setup is used. This block has the function of indicating which subsystems will run on the processor.

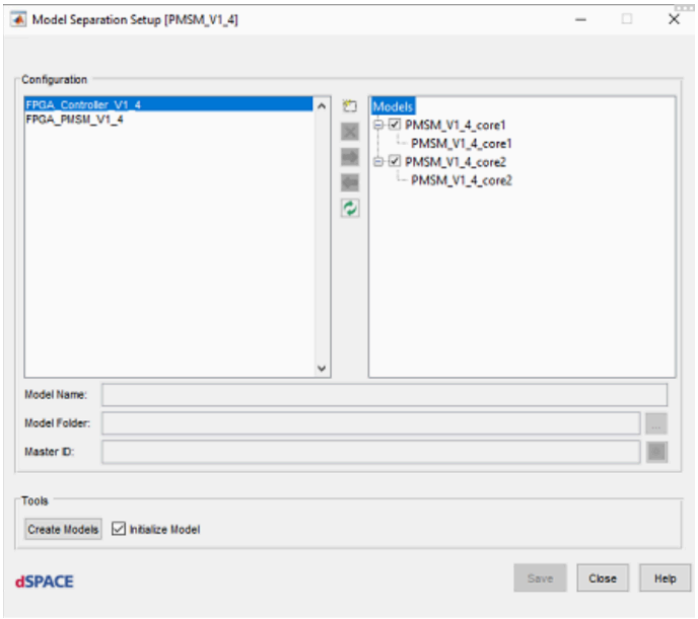


Figure 5.37 – Model Separation Setup

At the end of the configuration, on Configuration desk you will find the following screen:

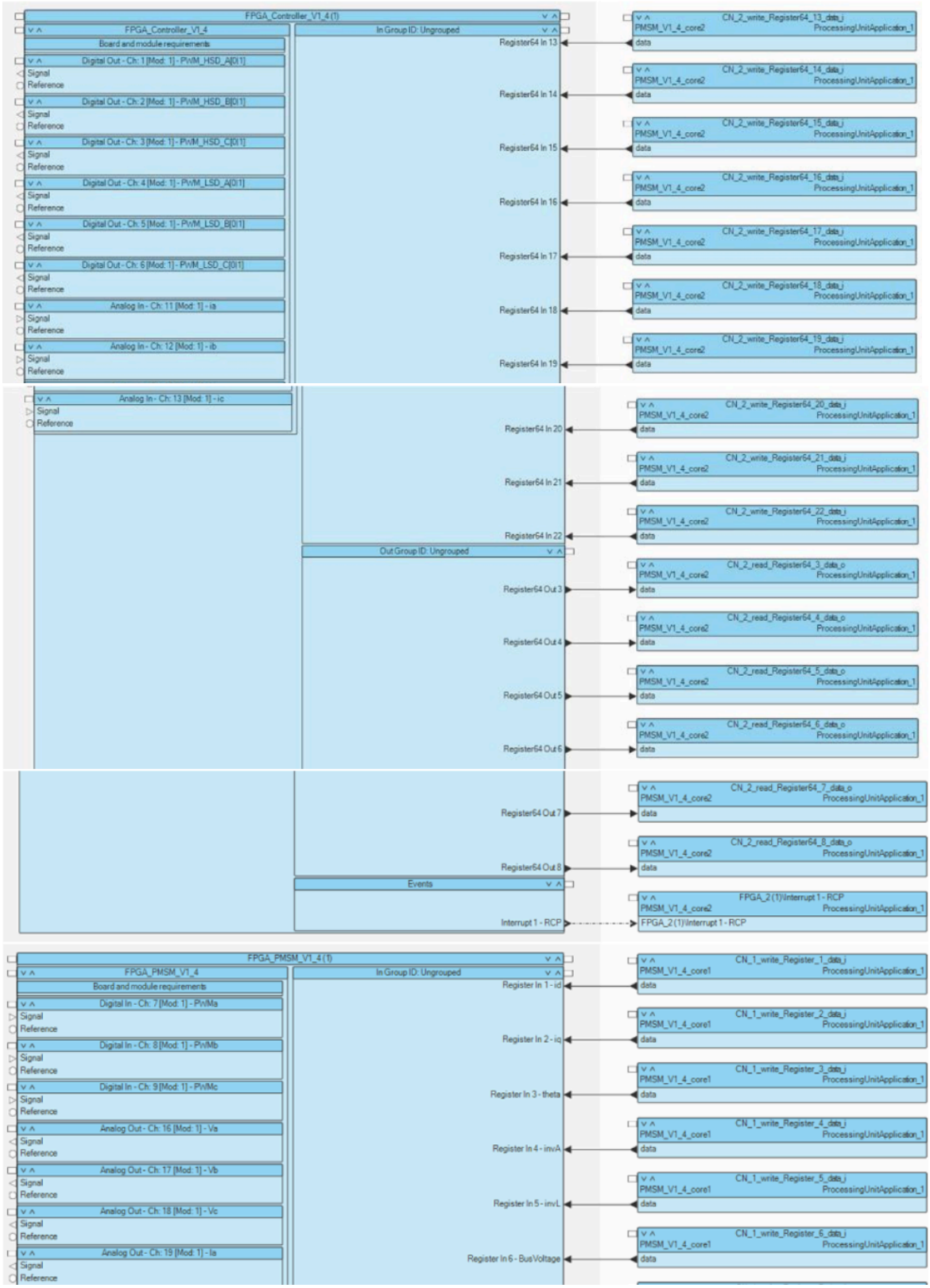


Figure 5.38 – Configuration Desk interface of the model

On the Multi Module screen you can see the information exchanged between the models running on the processor:

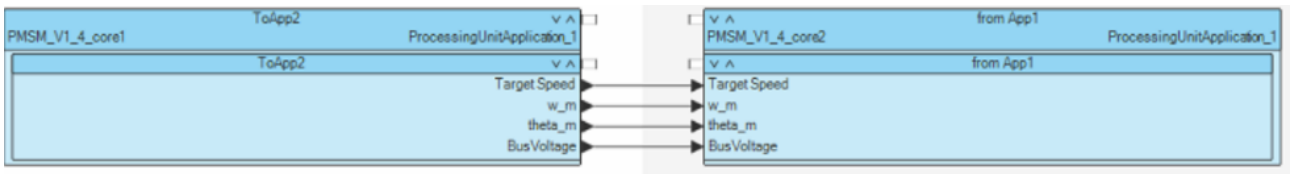


Figure 5.39 – Multi Module

Once you have achieved this configuration, you can import the simulator hardware that will be used and then build this new model. This operation will generate an extension file. sdf that will be inserted in the simulator so that you can run the application in real-time simulation

Figure 5.40 – Configuration Desk build

5.5 GUI

In this section will be presented those that were the graphical interfaces developed in such a way that you can control the application in real-time but above all you can make it as intuitive and accessible as possible.

5.5.1 GUI overview

The first graphical interface represents the whole model in its completeness:

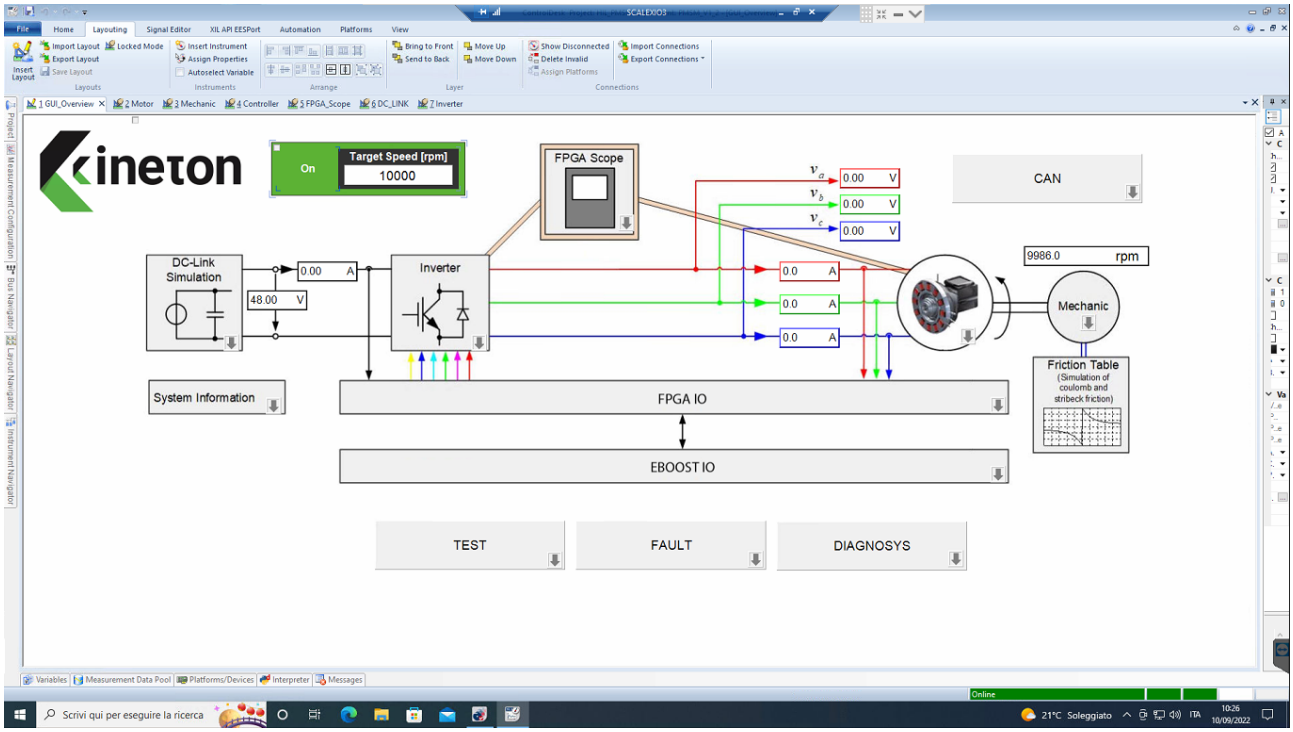


Figure 5.41 – GUI Overview

First of all, we can see in the upper left, the presence of an ON/OFF button from which you can turn on or off the Device Under Test (DUT) and also check the speed of the motor. In addition, as it changes, it is possible to see through a series of textboxes, the value of the supply voltage (V_{dc}), current (I_{abc}) and voltage (V_{abc}) in phase and mechanical speed of the motor(ω_m).
Finally, the template itself provides a quick link to the other layouts described below.

5.5.2 Electric model

This interface gives the complete vision of what is the electric model of our Permanent Magnet Synchronous motor (PMSM):

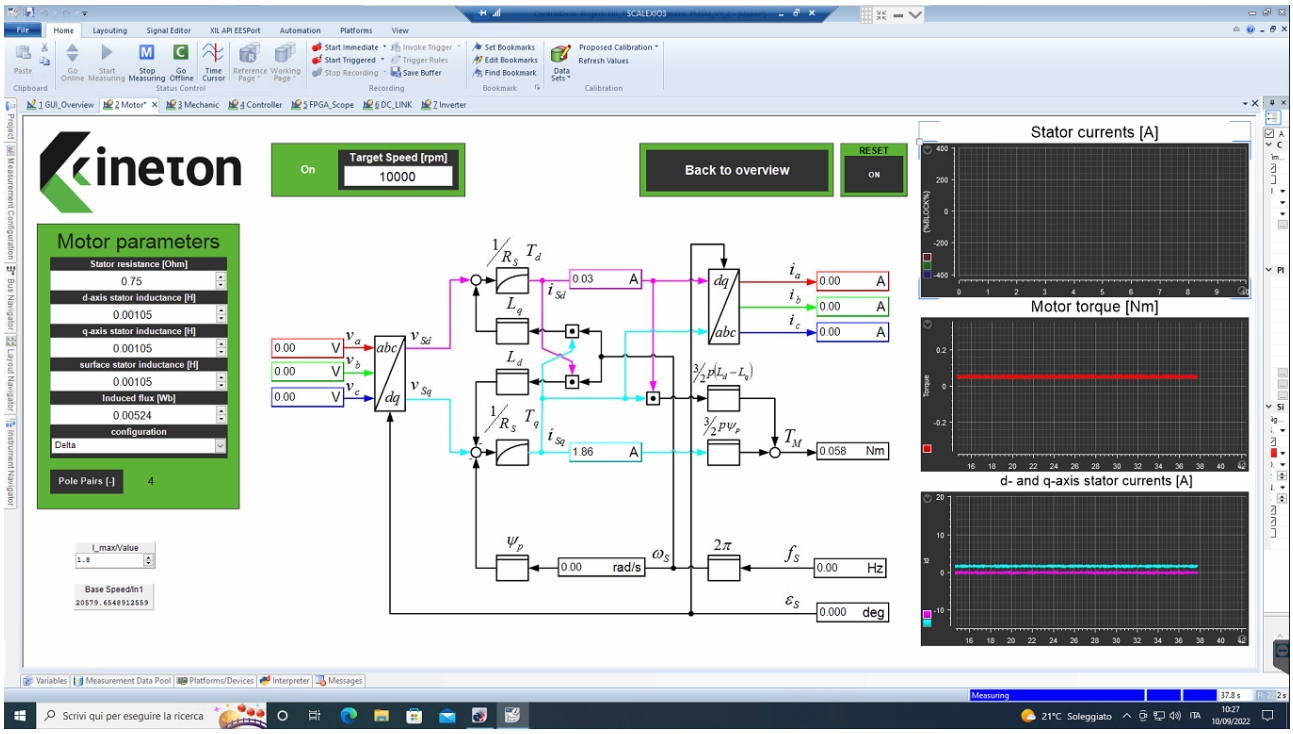


Figure 5.42 – Electric model of PMSM

As it has been described in the last interface, we can immediately notice the presence of a button ON/ OFF and a textbox able to insert and control the speed of the engine. Moreover, on the right side we have a link to the general overview layout and a Reset button.

Moreover, on the left side we can change in real-time the characteristic parameters of the motor as for example the stator resistance, the inductances or the flow. In addition, you can decide whether to use a triangle or star engine configuration.

As in the last interface, it is also possible to view the changes thanks to the displays and visualize their trend thanks to the presence of three graphs on the right side. Starting from the first graph at the top, it is possible to visualize the phase currents in three-phase steady-state (I_{abc}), the torque (T_L) and the corrects of the rotating frame (I_{dq}).

5.5.2 Mechanic model

This interface gives the complete vision of what is our mechanic model of our Permanent Magnet Synchronous motor (PMSM):

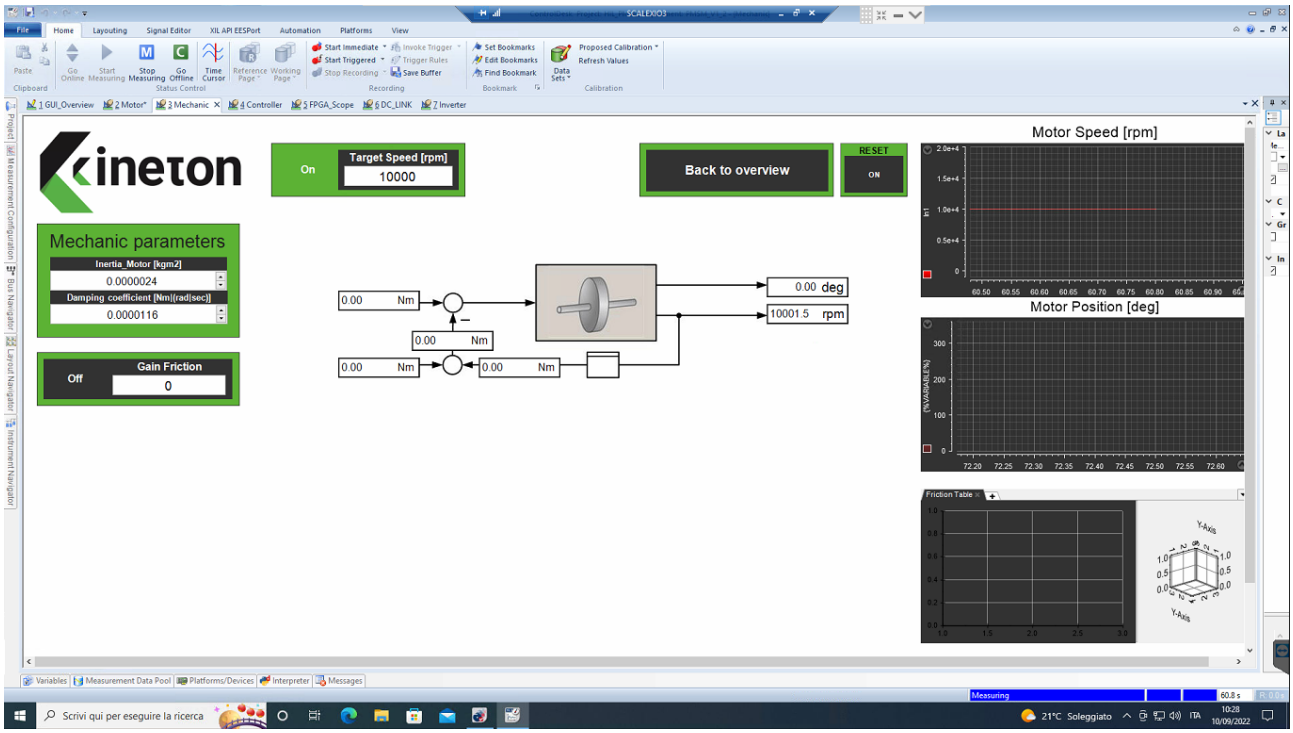


Figure 5.43 – Mechanic model of PMSM

Also in this layout we can find the same buttons present in the previous interfaces: ON/OFF, Reset, Target Speed and 'Back to Overview'.
On the left side we find textboxes that allow us to change in real-time the mechanical parameters of the engine such as the inertia or the damping coefficient.
In the center we can see from the appropriate display the change of the values of the model, while on the right side we find three graphs that represent, from the top, respectively the speed of the engine and the mechanical position.
Finally, you can activate a gain friction in order to insert in the model a torque that depends on the speed. However, it has not been implemented in the model.

5.5.4 Inverter

This interface gives the complete vision of the inverter:

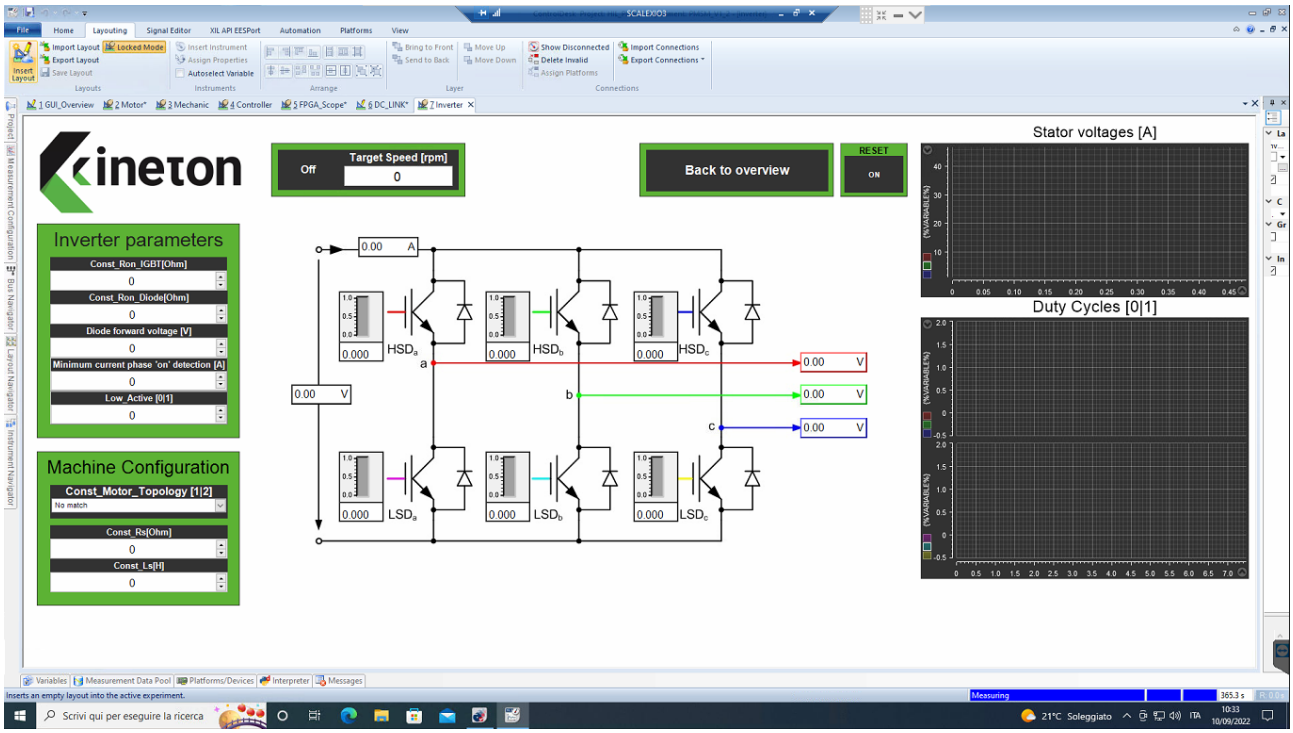


Figure 5.44 – Inverter

Also in this layout we can find the same buttons present in the previous interfaces: ON/OFF, Reset, Target Speed and 'Back to Overview'.
On the left side we can find textboxes where you can change the inverter parameters in real-time.
In the inverter diagram it is possible to see through the appropriate displays the trend of the PWM on the individual transistors and the phase voltages for each branch generated.
The graphs on the right side represent the supply voltage and the duty cycles that control the transistors.

5.5.5 Controller

In this section you can see the implemented controller structure:

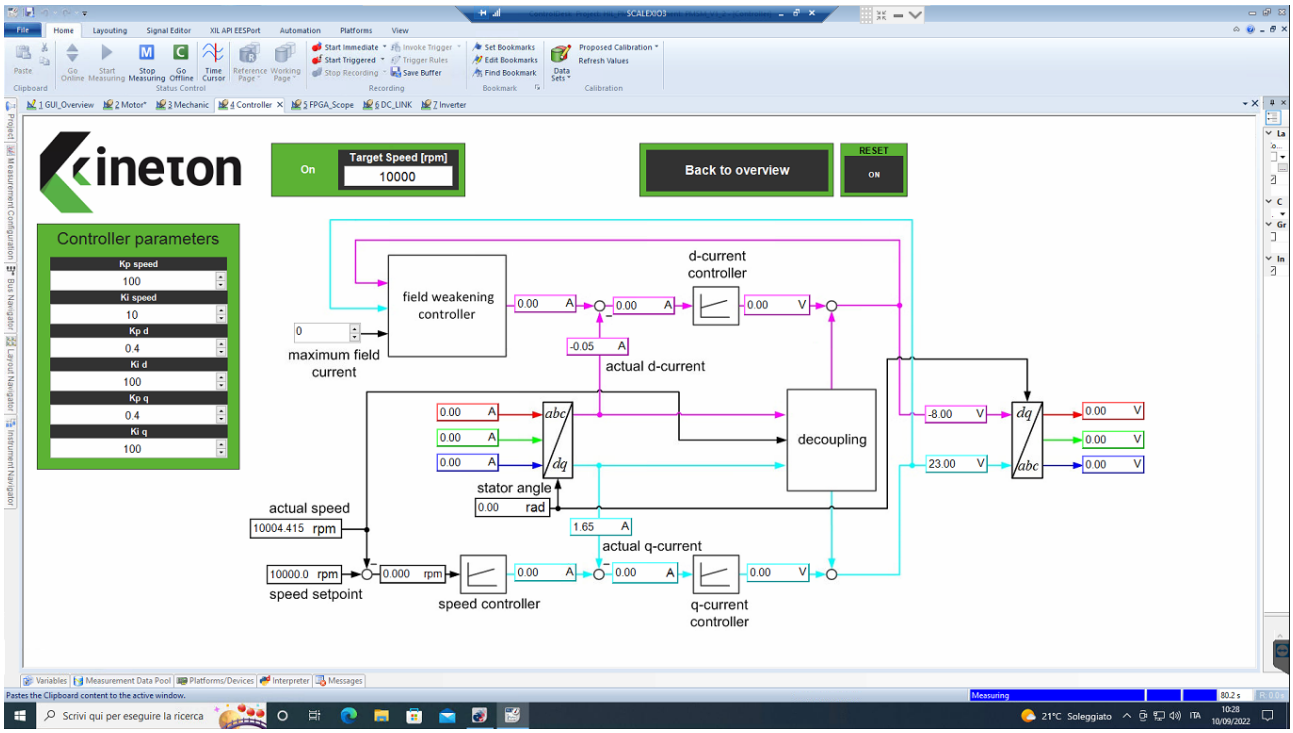


Figure 5.45 – Controller

We can always find, as it is in every layout, the ON/OFF button, the RESET, 'Back to overview' and the textbox speed control.
On the left side it is possible to change in real-time the proportional and integral coefficients of the 3 PIs of the controller.
In addition, you can view the instantaneous trend of the most important signals of the controller.

5.5.6 DC Link

This layout has been structured in such a way as to enable or disable the DC Link simulation, thus simulating a real or ideal power phase. However, this feature has not yet been implemented in the model

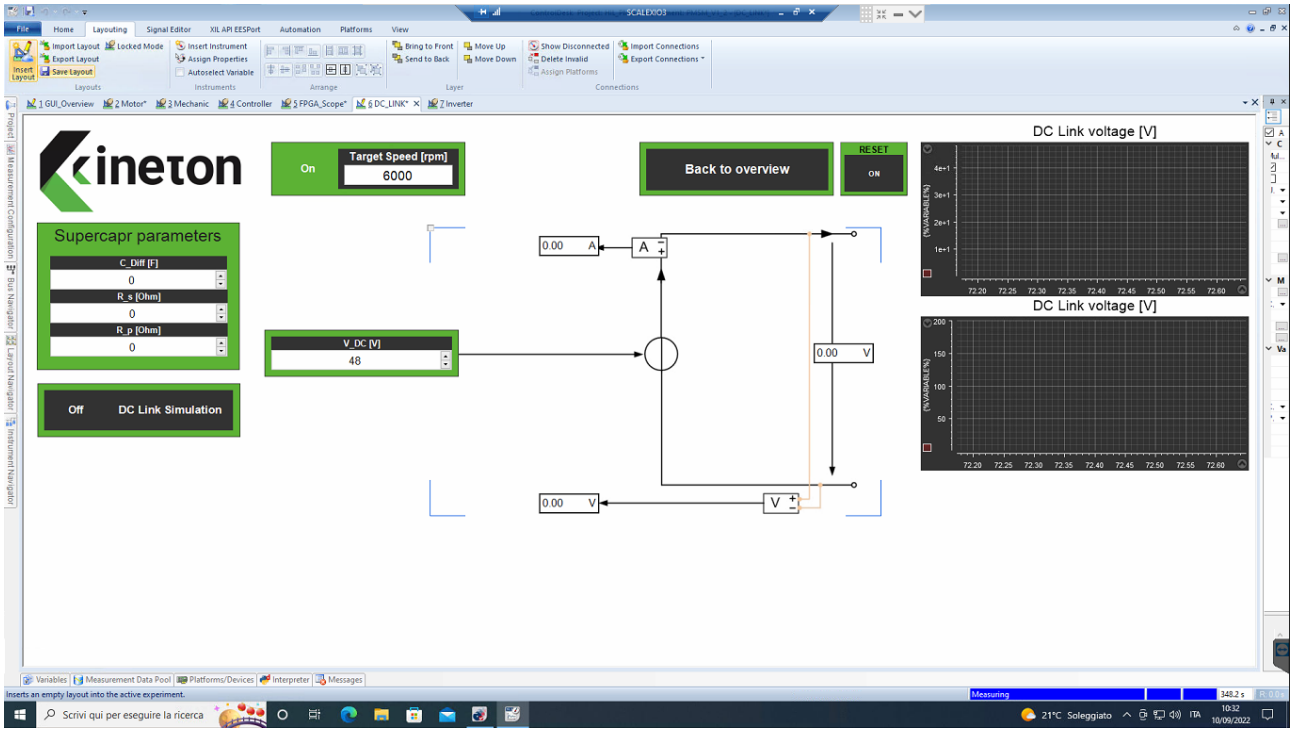


Figure 5.46 – DC Link

5.5.7 FPGA Scope

In this last layout, there is the possibility of displaying signals in the FPGA, characterized by faster dynamics than the processor signals. For this purpose, therefore, the FPGA scope is present in the model.
In particular, it is possible to see the trend of voltages and phase currents, currents and voltages in the dq regime, PWM, mechanical speed and mechanical position.

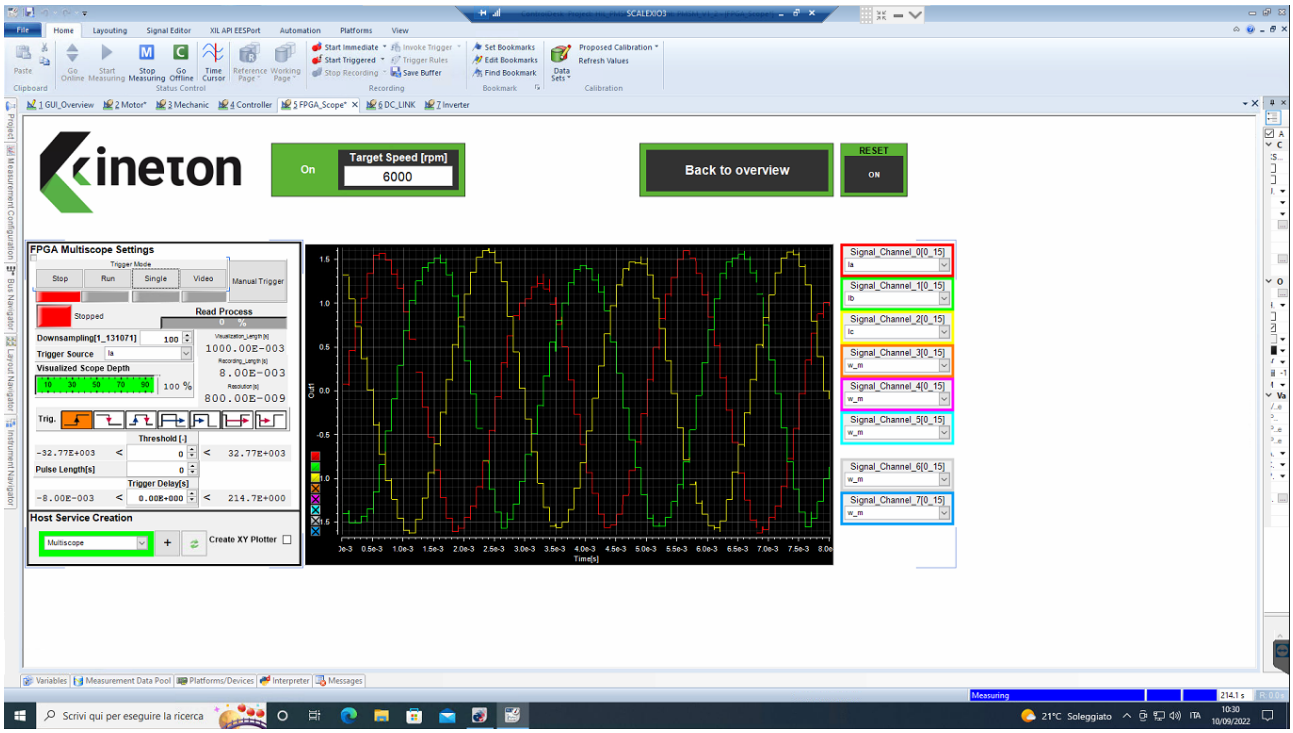


Figure 5.47 – FPGA Scope

6. Discussion of the Results

6.1 Version 1.1

The aim of the thesis, besides validating the model in its completeness, reaching high speeds and good performance, was also to make it as malleable and parameterizable as possible.
To achieve the desired results, the following values have been chosen:

```
CON.Kp_spd = 100;
CON.Ki_spd = 10;
CON.Kp_d = 0.4;
CON.Ki_d = 400;
CON.Kp_q = 0.4;
CON.Ki_q = 400;
MOT.Rs = 0.75;
MOT.Ld = 1.05e-3;
MOT.Lq = 1.05e-3;
MOT.I = 2.4018552467e-06;
MOT.P = 4;
MOT.Ke = 3.8;
MOT.phif = (MOT.Ke)/(sqrt(3)*2*pi*1000*MOT.P/60);
MOT.Bm = 11.603710493e-06;
MOT.I_max = 1.8;
MOT.configuration = 1;
MOT.FPGA_step = 8e-9;
```

Figure 4.1 - Parameters

Following the logical thread that was taken in the last chapter, at a first was tested the electric and mechanical model in the Synchronous Permanent Magnet motor (PMSM).

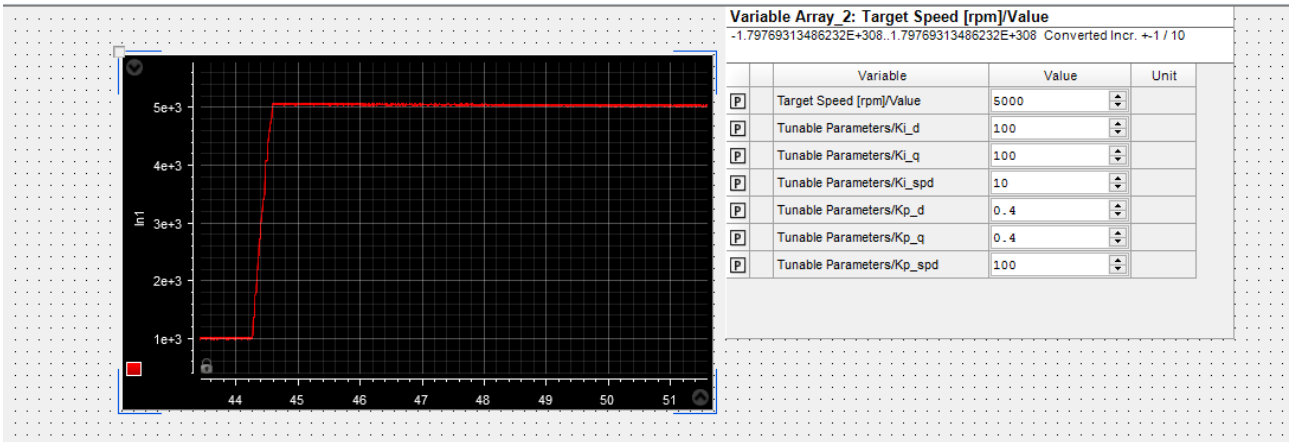


Figure 5.2 – 5000 rpm Target Speed

By inserting a $targetSpeed = 500\,rpm$,we can see in the figure (6.2), how the engine follows its target and reaches the desired speed.
In order to make the response smoother and less noisy as possible, the integral coefficients of the controller ki_d and ki_q have been changed in real-time, and set at:

$$ki_d = ki_q = 100$$

In the second example below, a $target\ speed\ =\ 12000\,rmp$ has been set:

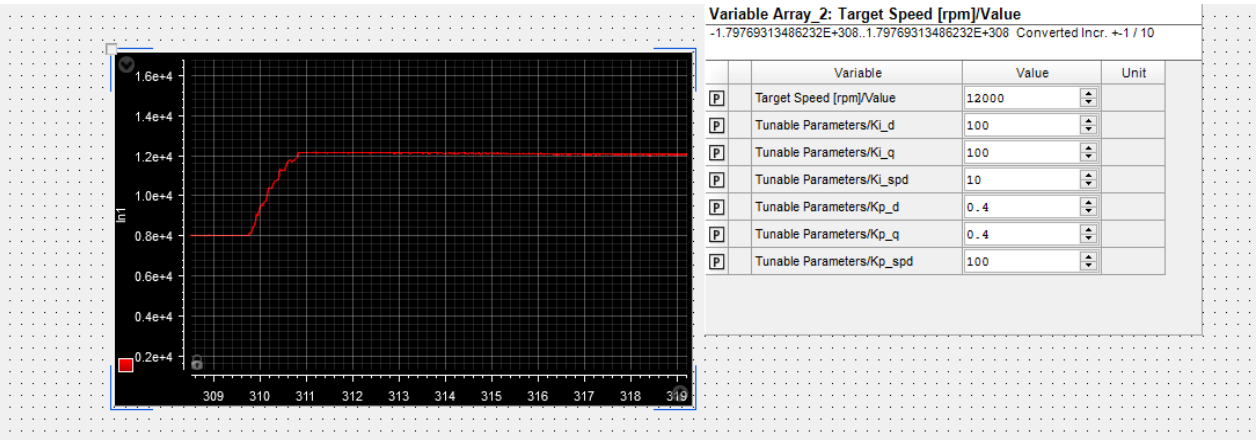


Figure 6.3 – 12000 rpm Target Speed

Finally, remembering the control logic implemented for the target speed, a value higher than the base speed of the engine was added. Therefore, we can see that it does not follow the set target but it saturates to the value of the base speed instead:

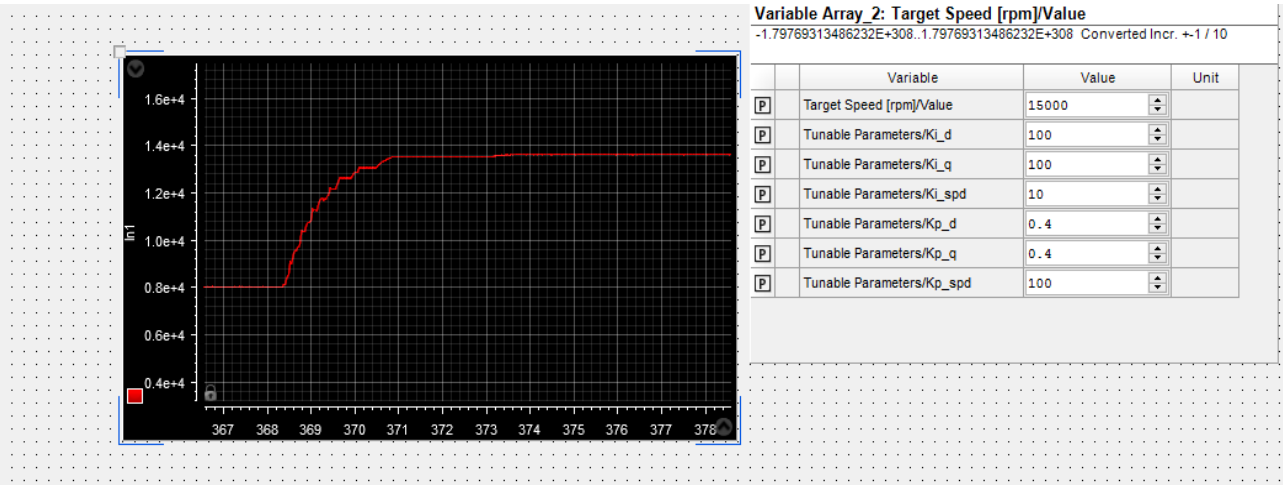


Figure 7.4 – Saturate Base Speed

6.2 Version 1.2

As it was anticipated in the last chapter, version 1.2 is to be considered as an upgrade of the model first presented. After having correctly implemented and tested the electric and mechanical model of the PMSM, the Clarke & Park transforms have been added to the model.

Figure 8.5 – Clarke & Park transform through FPGA Scope (5000 rpm)

In the figure (6.5), through the use of the FPGA Scope, we can see the course of both the i_{abc} currents and the v_{abc} voltages with a target speed equal to $targetSpeed = 5000\ rpm$.
Below will be given a similar example but with a target speed set to $targetSpeed = 10000\ rpm$.

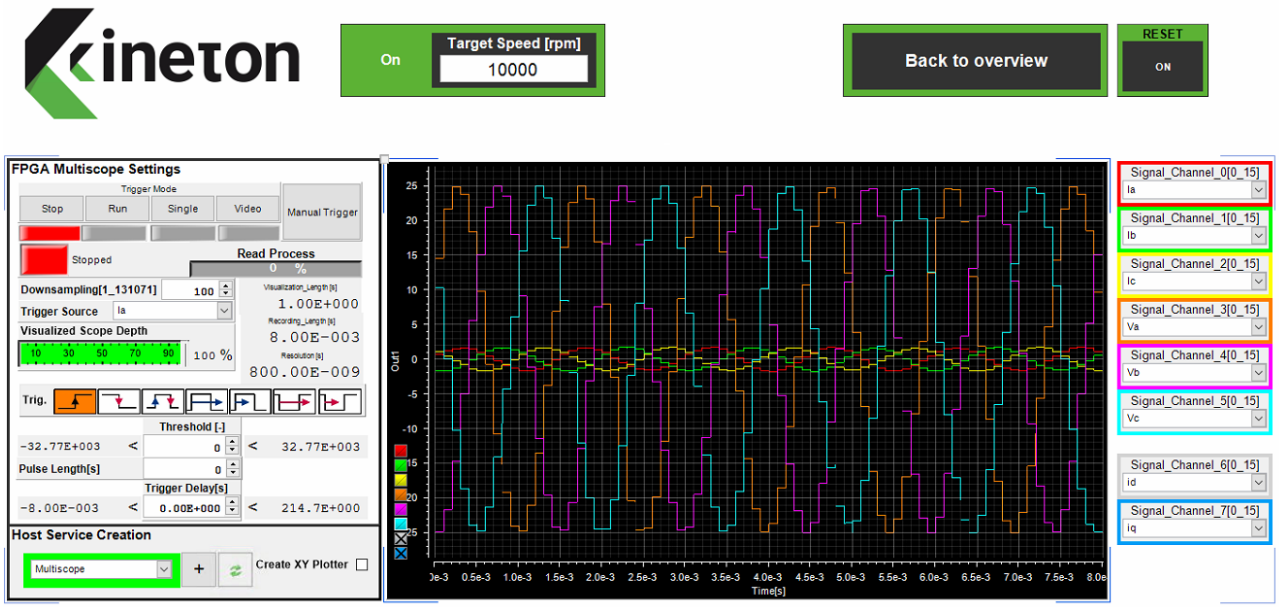


Figure 9.6 – Clarke & Park transform through FPGA Scope (10000 rpm)

Moreover, through the use of specific RTI blocks, three 'Analog_out', it was possible to see the trend of the v_{abc} voltages thanks to the use of an oscilloscope

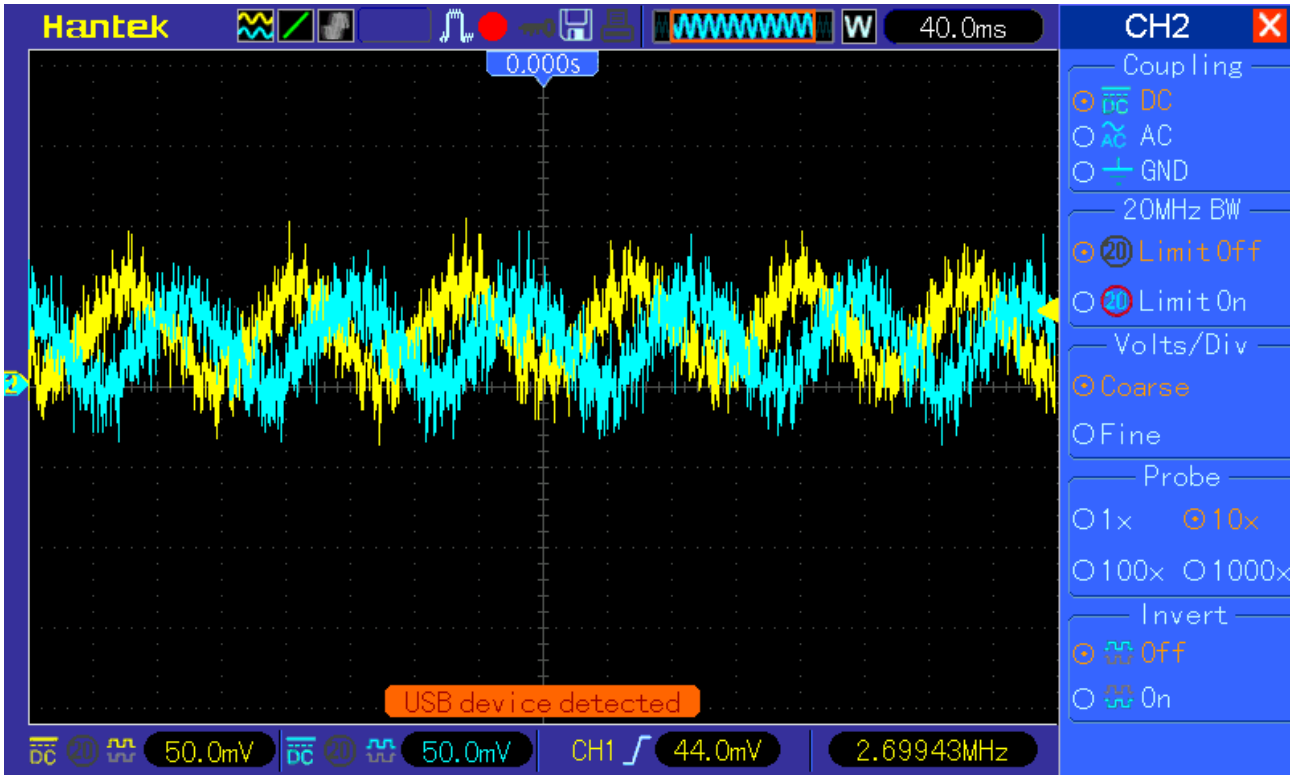


Figure 10.7 – Clarke & Park transform through Oscilloscope

6.3 Version 1.4

In this latest version 1.4, which it has been explained in detail, the controller we developed in FPGA has been deleted and replaced with the controller already implemented and located in the dSpace matlab libraries. In this way it was possible to test the final model in its completeness.

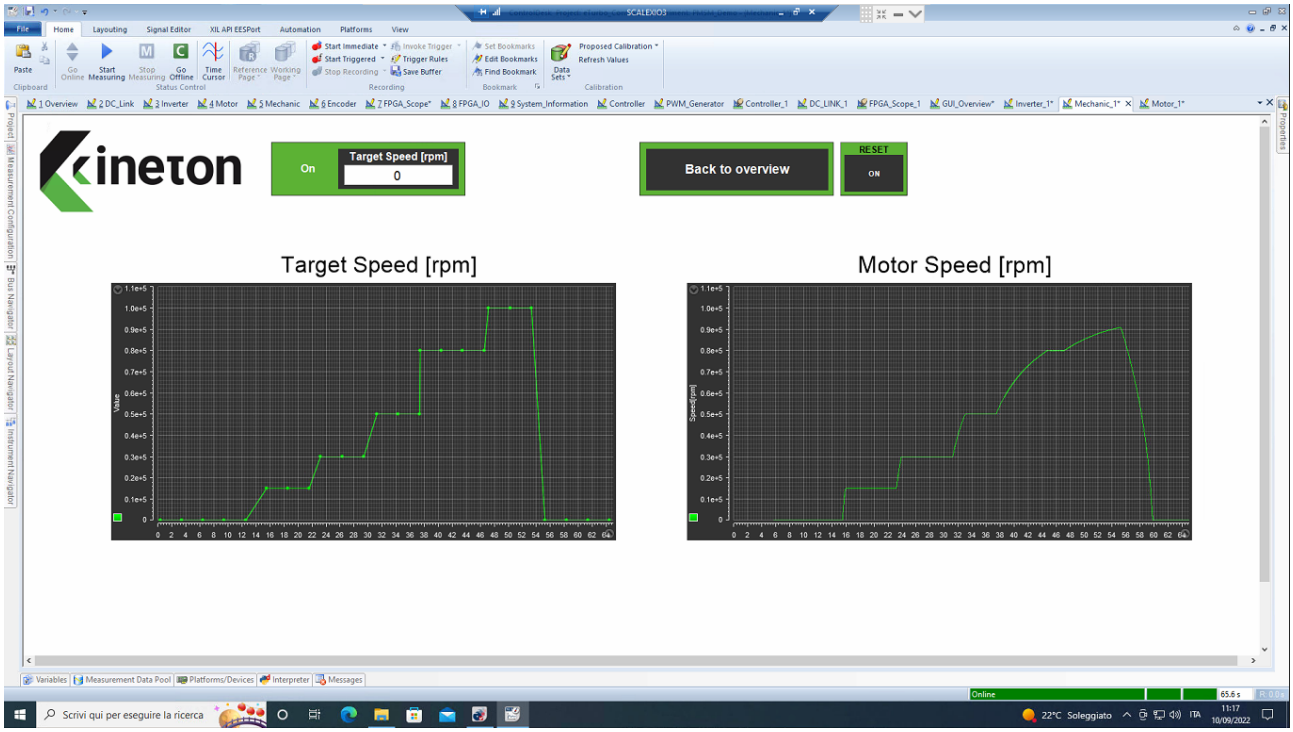


Figure 11.8 – Target Speed (rpm) vs, Motor Speed (rpm)

In the figure (6.8), we can notice from the two graphs, as the course of the speed of the motor appropriately follows the values of the set speed of target from an appropriate textbox situated in top left. In addition, we can note that when the Reset button is switched on, the engine speed returns to a value of 0 rpm (engine off).

Next, an overview of the engine's electric model is shown, along with three graphs representing stator currents (i_{abc}), Motor torque (T_L) and currents in the orthogonal rotating reference frame (i_{dq}).

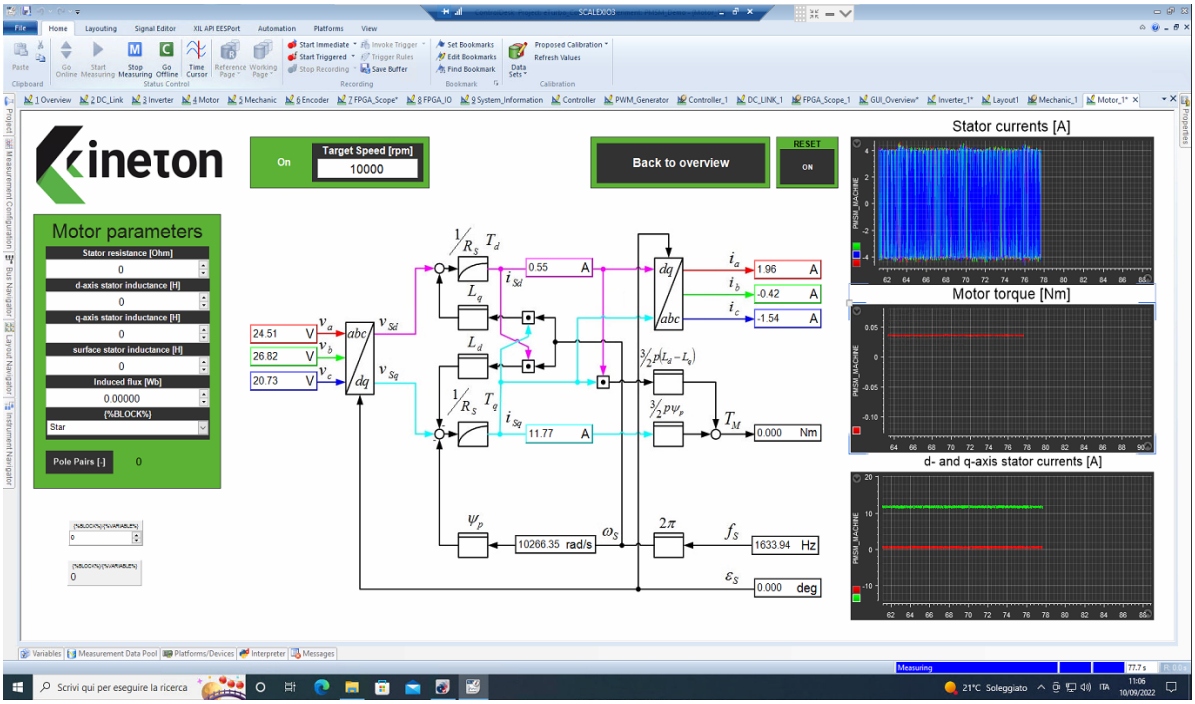


Figure 12.9 – Electric model of PMSM simulation

Finally, as for the inverter model, the following image shows the trend of the values and on the left side are represented the graph of the supply voltage ($v_{dc} = 48V$), and the PWM.

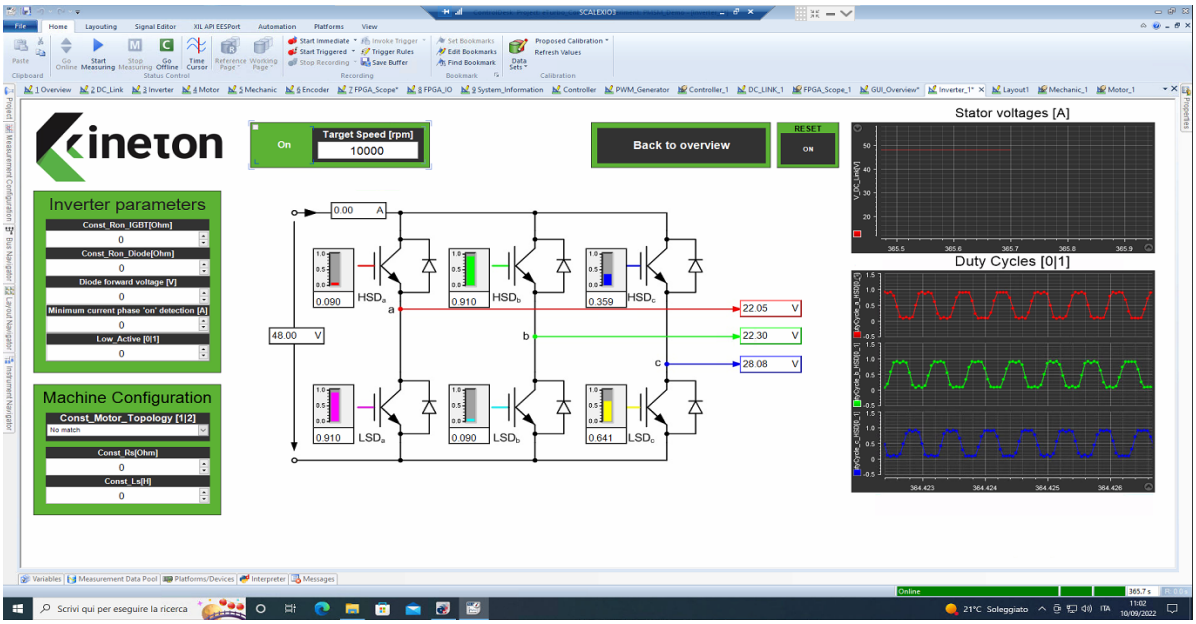


Figure 13.10 – Inverter model simulation

7. Conclusion and future developments

Nowadays, Hardware-in-the-loop (HIL) techniques are increasingly in demand as this type of simulation allows testing the systems both in its intended modes and under dangerous situations. Moreover, this type of simulations run at real time speed and perform I/O with the Device Under Test (DUT); in this way the test item believes it is operating as a component of a real systems in its operational environment.
As it has been extensively discussed in Chapter 5, considered the complexity of the project itself, several models have been created with structures of increasing difficulty.
In this way, it was possible to test one by one the electric and mechanical model of the Permanent Magnet Synchronous motor (PMSM) and the Clarke & Park transforms.
Once these models have been successfully tested, the final model has been created by using a FOC control unit as a device under test (DUT), which was already implemented and located in the dSpace Matlab libraries.
Thanks to the implemented model and the parameterization used, it was possible to achieve excellent system performance both in term of response to noise and for the achievement of high speeds.
Moreover, having operated outside the flux-weakening, it has been possible to spin the motor approximately up to 13000 rpm.
Therefore, considered the impossibility of having a real control unit to be used as a device under test, one of the next steps of my project will be then to replace the dSpace controller with a real control device.
In this way it will be possible to make a comparison with the model tested during my thesis and to visualize the differences both in terms of performance, speed achieved, parameters used and furthermore to adapt the model according to different and specific requirements.
Therefore, last but not least, a future job will be to verify and test the response of this model in such a way as to reach the same objectives even by using another device under test (DUT) different from the one used in this project.