Politecnico di Torino

Facoltà di Ingegneria Meccanica

Corso di Laurea Magistrale in Ingegneria Meccanica percorso Automazione

Design and Testing of Hardware in the Loop Test Bench for Permanent Magnet Synchronous Motors controlled with Field Oriented Control Technique

Candidato: Emanuele Frasca

Relatore: Prof. Stefano Mauro

Anno accademico 2021-2022

INDICE

Abstract

This Master Thesis is the results of an internship in Teoresi SpA that provides the Hardware and the Software used to build an Hardware In the Loop (HIL) Test Bench for Permanent Magnet Synchronous Motors (PMSMs). The aim of this work is the design and the testing of the HIL Test Bench. In the Test Bench the HIL 602+ Device by Typhoon HIL is used to real-time simulate the physical behavior of the electric machine. The real-time Linux operating system is used to run the Field Oriented Control (FOC) algorithm and is operated from LabVIEW. Both Typhoon and LabVIEW software have been used to create an user-friendly Human-Machine-Interface (HMI) to run and monitoring in real-time the system simulation. Particular attention has been paid in the application of the field-oriented technique including the possibility to achieve field-weakening conditions. Another important aim has been the testing of the performance needed by the system in terms of scan-time and how they can be matches form the device performances. To better understand the logic flux that has been followed here is reported a short list of the principals step of the all development:

- Hardware in the loop technique in servomechanism applications;
- Permanent magnet synchronous motor working principals;
- Field-Oriented control strategy for interior PMSM;
- Design of the system in Simulink;
- Export, manipulation and validation of the C-code control algorithm;
- Virtual and real-time simulation of the open test bench;
- Loop-closing with external real-time OS;
- Closed-loop Simulation;
- Conclusions and future developments.

1. Field Oriented Control technique for Permanent Magnet Synchronous Motors

1.1. Introduction

In the recent years many industrial areas are experiencing a total renovation in terms of efficiency with the global challenge to reduce consumptions and automatization of all the electromechanical devices such as the autonomous drive implementations. For these reason development and control system techniques are nowadays one of the most required areas of the global research. This means, of course, that more and more frequently new electrical, electronic, electromechanical, and power electronic systems are developed, and new control techniques are improved. In this thesis just, a small part of these objects is involved but the sector is a huge and borderless area of knowledge.

Another important aspect to consider is the growing willing to reduce the waste also during the phase of production and testing of new technologies. Reducing the number of wasted prototypes not only impact directly on the consumptions but also nowadays is possible to accelerate the entire development process introducing a massive usage of simulation. To this proposal the well-known "Hardware in the Loop" Technique is more and more used to replace the old real physical test. Shortly we can say Hardware in the loop (HIL) is a simulation technique used in the development and testing of complex real-time embedded systems. In HIL Simulation the complexity of each dynamic system is replaced by a mathematical representation which can be differently close to the real plant emulated. To recap the HIL technique can be profitable in terms of:

- Costs: it's possible to avoid many unnecessary physical tests, it becomes possible to replace a parameter of the system without replace all the physical component of the plant but also because it allows to test the plant in failure condition without risks.
- Duration: as we mentioned the process of testing can be done also without any physical prototype and this allows to avoid any production slowdown or issues.
- Safety: this because the testing is based only in a mathematical representation, also the possibility of testing any failure condition without risk can be important to also evaluate those conditions which can't be tested for real.
- Feasibility: also, is possible to eliminate in the process many restrictions in terms of accessibility to the real condition: for example, testing a large range of features and verify

before any resource has been wasted which one can be more useful for our purpose (i.e. new material is often implemented in this way).

In Figure 1.1 is reported a logical scheme of a HIL testing bench, in particular this scheme will be very close to the one presented in the results part of this thesis

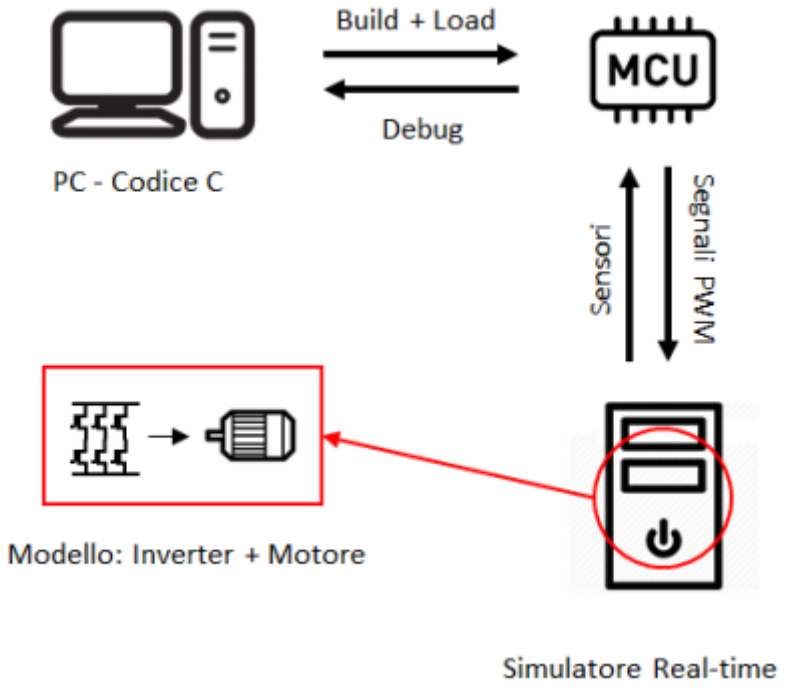

Figure 1.1 Logical Test Bench used in a Hardware in the Loop Simulation[1]

1.2. Working principles of a Servomechanism

Before entering a general classification of electrical machines and their control techniques it's important to introduce what are the working principles of what is commonly called a "servomechanism". Shortly it's possible to say that a servo (servomechanism) is an electromagnetic device which is responsible to convert electricity into precise controlled motion by use a negative back mechanism. The term is correctly used only to systems where the feedback or error-correction

[1] E.Giammello (2018/2019) - Simulazione e test HiL del controllo di azionamenti elettrici in ambienti MATLAB e PLECS – Politecnico di Torino

signals help control mechanical position, speed, torque, or any other measurable variables. For example, an automotive power window control is not a servomechanism, as there is no automatic feedback that controls position – the operator does it by observation. By contrast a car's cruise control uses closed-loop feedback, which classifies it as a servomechanism.[2]

Motor, Gear and Load are the three main objects to visualize when you want to analyze a servomechanism. But the most important and discriminating object that, as said before, identify a device as a servomechanism is a "feedback mechanism" which is responsible to "read" the physical signal coming from the load and transduce it into a signal "readable" by the control unit which can modulate its output according with the measured feedback received by the transducer. Commonly these devices are also called "sensors" although they are different objects because sensors measure a physical quantity and converts them into a scale of measure depending on the typology of sensor, for example a litmus paper convert a value of ph (acidity or basicity of a chemical substance) into a color gradation. So, between the two objects it must be interposed a third object which can be named "transmitter" that is able to convert two different types of signals. To resume, a Transducer is made of a sensor and a transmitter. From now on however, clarified the difference, the two terms will be used as synonyms.

To better clarify the difference, introduce between a close-loop system and an open-loop one (where the command signal does not depend on a feedback signal) let's introduce the simplest existing electric motor dynamic:

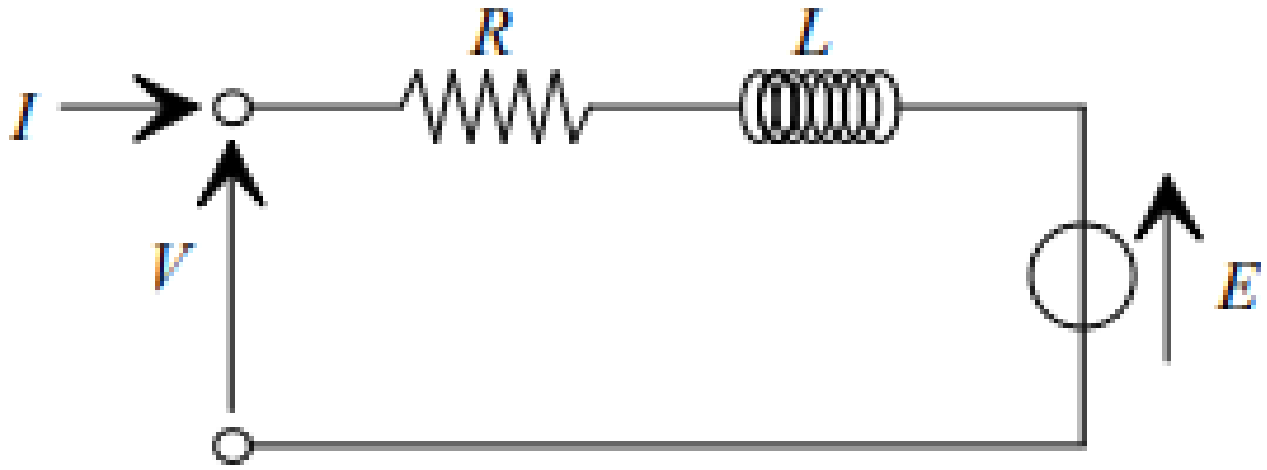

Figure 1.2 RLC circuit for a DC motor.

[2] https://en.wikipedia.org/w/index.php?title=Servomechanism&oldid=1082552982

The circuit in Figure 1.2 is described by these equations:

$$\begin{cases} V(t) = RI(t) + L\dfrac{dI(t)}{dt} + E(t) \\ E(t) = K\omega_m(t) \\ \tau(t) = KI(t) = J_m\dot{\omega}_m(t) \\ \dot{q}(t) = \omega_m(t) \end{cases}$$

Where E(t) is the Back-Electromotive force (bemf) in Volt, $\tau(t)$ the torque generated by the motor in Nm, J_m the mechanical inertia of the load-motor system in Kg*m^2 and the two K are respectively the Electromotive force-constant [Vs/rad] and the Motor torque-constant [Nm/A].

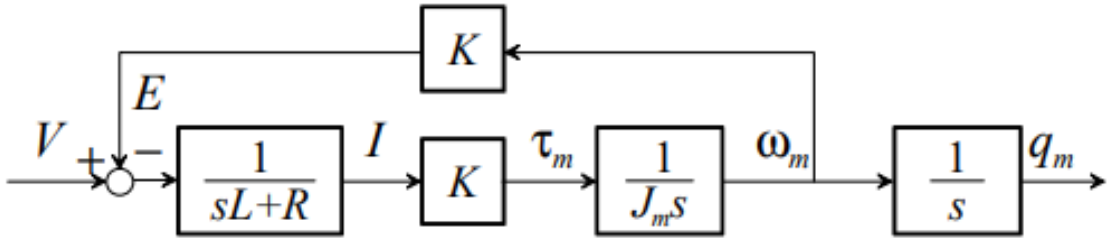

Figure 1.3 This is a schematic block diagram of the transfer function between the input V and the output q_m.

Let's suppose now that is possible to measure the current value (using an Ammeter) the scheme can modify into the one represented in Figure 1.4. It's possible to notice that the first summatory block produce a value that represent the "error" of the system so it's reasonable to expect to see this value to be zero when the system works in regime mode.

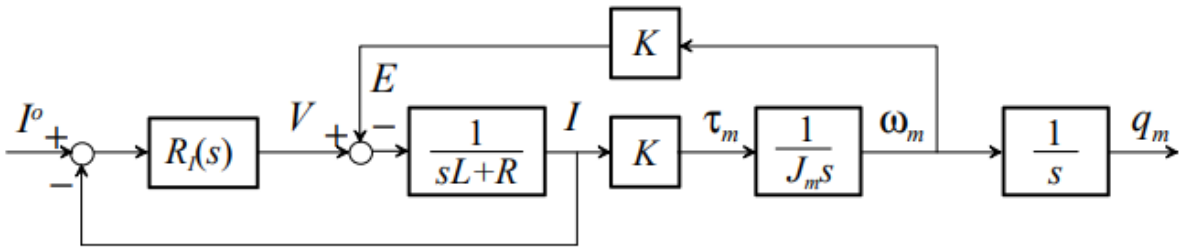

Figure 1.4 Diagram block of a current controlled electric DC motor.

Typically, this error is controlled by a controller block. As it will be discussed in the following chapters, the mostly commonly known controller is Proportional-Integrative-Derivative controller, as known as PID controller where each word represents a branch of control as represented in Figure 1.2 RLC circuit for a DC motor.

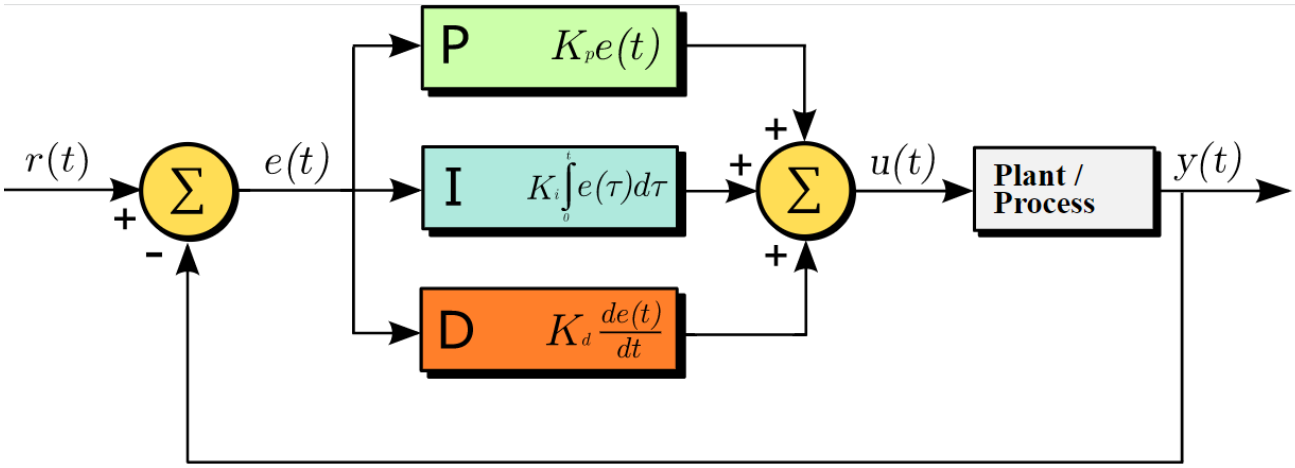

Figure 1.5 A block diagram of a PID controller in a feedback loop. r(t) is the desired process value or setpoint, and y(t) is the measured process value.[3]

Mathematically the PID control unit is described as:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{\partial e(t)}{\partial t}$$

Where K_p, K_i and K_d are respectively the proportional, integrative, and derivative gains, e(t) is the error and u(t) the controlled variable. Also, usually the equation above can be written in this way:

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau)\,d\tau + T_d \frac{\partial e(t)}{\partial t} \right)$$

[3] https://en.wikipedia.org/w/index.php?title=PID_controller&oldid=1090592841

Where T_i and T_d can be associated to a real physical meaning:

- $\boldsymbol{K_p T_d}$ gives information about how fast the system is to approach the set-point.
- $\boldsymbol{K_p/T_i}$ gives information about how precise the output can be related to the reference (above or below).

Exactly as it has been shown for the motor also the load can be described with mathematical model, which can be differently sophisticated according with how much you want to be close to the reality, but also depending on the size of the system. Typically for this kind of system the mathematical approach to the model is to consider the system to be a lumped-element model. In Figure 1.6 is shown a free body diagram of the mechanism the usually can be used to approach motor-gear-load system as it is an electric motor application.

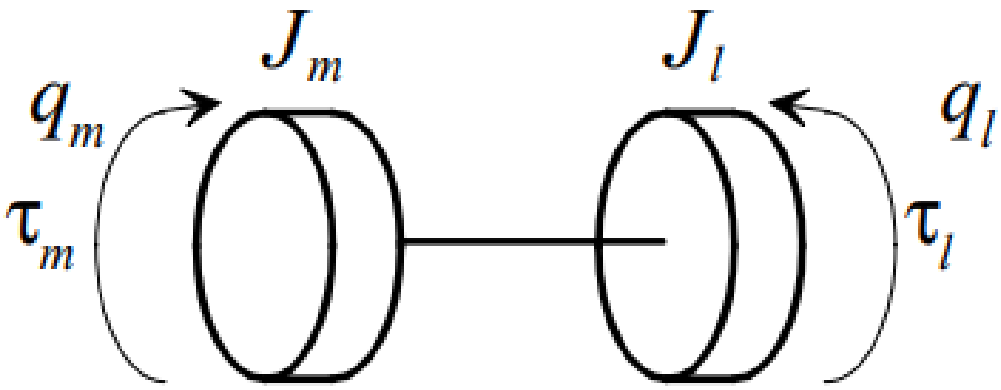

Figure 1.6 Free body Diagram of Motor, Gear and Load with torques and angular positions highlighted.

The equations below describe the entire system mathematics, the first equation will be the most important used in this thesis because it expresses the relation between torque (which is the input given by the electric motor) and the cinematic quantities as they are angular position, speed and acceleration that will be involved in the feedback.

$$\begin{cases} J_m \ddot{q}_m + B_m \dot{q}_m = \tau_m - \tau_{lm} & ;for\ the\ motor \\ J_l \ddot{q}_l = n\tau_{lm} - \tau_l & ;for\ the\ load \\ q_m = q_l n & ;for\ the\ gear \end{cases}$$

1.3. Electric Motor general classification

Electric Machines are electro-mechanical devices which purpose is the conversion of a shape of energy into another based on the laws of magnetic induction (as Faraday-Neuman-Lenz law) and electromagnetic force (as Lorenz's force).
In the large field of electric machines, it's possible to classify: "static machines" when used to convert electric energy into a different shape of it, in this classification we can mention the transformer, the most know and wide used. "Dynamic machines" (or rotating) are used to convert mechanical energy (usually kinetic) into electric energy and vice-versa. In the first case it's generally called "Motor" in the other one "Generator". Between the rotating electric machines, it is also possible to distinguish 2 main other classification criteria: the shape of current utilized which can be DC or AC and the ration between the angular speed of the rotating magnetic field and the rotor itself, in this case when they are synchronized, we are talking about "Synchronous" machine, vice-versa "Asynchronous" one in the opposite case. In this thesis an "Synchronous Machines is discussed, so, now a short recap of its properties and features is proposed to understand also which are the advantages and the disadvantages and how they can be related with its applications.

1.3.1. Synchronous Electric Motors

Every synchronous machine is composed by a rotor (moving part) fed by a DC excitation and by a stator (stationary part) where the alternating current flows through the armature three-phase windings. In some case, one of more pairs of permanent magnets can be mounted on the rotor for producing a magnetic field that is able to produce a rotating magnetic field the is the responsible of rotation of the rotor itself. Synchronous machines are generally used where the synchronism of the magnet field and the mechanical rotation realize very stable and accurate speeds: an example is the alternators (machines are generally called in this way when they work as generators of energy, motor when the use the energy) that is the main essential machine in electricity production. As their name suggests, Synchronous Machines works only when the rotating magnetic field speed is equal to the angular velocity (this is called Synchronous Speed) this create some trouble in the Torque-Speed characteristic that is presented as in the following Figure 1.7. It's easy to understand that the starting process of a machine become difficult because the starting Torque should be null at the starting speed that is very far from being null.

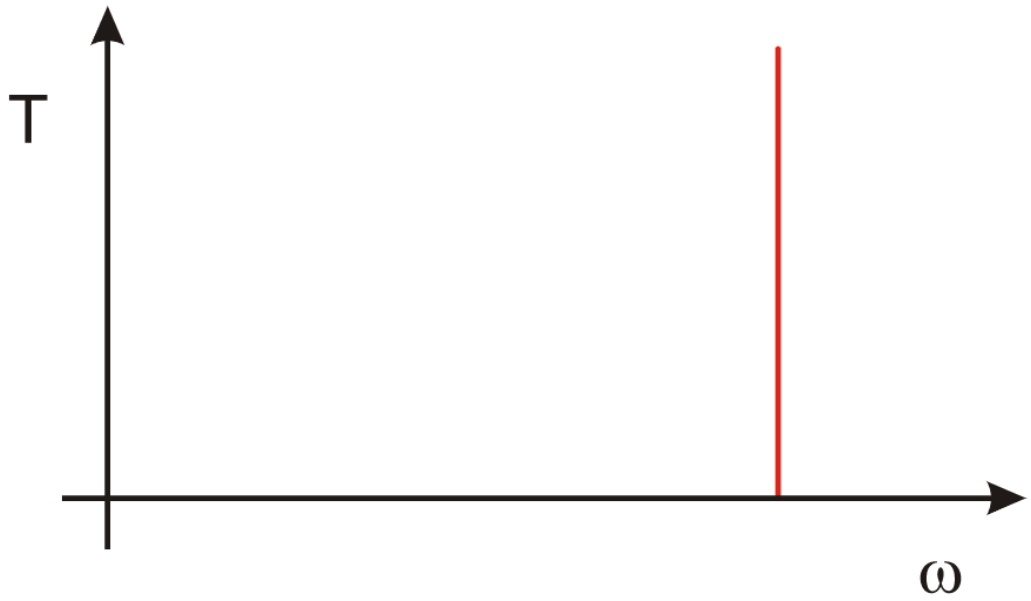

Figure 1.7 Speed-Torque characteristic of a Synchronous Motor: there's an evident absence of dependency on the speed this cause issues during the starting phase.[4]

Due to the starting issues these motors have only recently been used in variable speed application, as it's shown in the precedent Figure 1.7 starting curve is completely impulsive. This means that the starting process must be supported by another asynchronous motor. Today, the growth of the power electronics simplified a lot these starting issues, since it's possible to regulate the fed tension and the frequency at the same time it becomes easier to start from a null value of speed (frequency) and makes it increase continuously accelerating the motor. This is possible thanks to a device known as inverter that can be controlled by many different strategies that will be shown in the next paragraph.

Permanent Mounted Synchronous Motors (PMSMs) guarantees optimal performance, and, for this reason, it is implemented in recent automotive Electric Vehicles and Hybrid-Electric Vehicles applications. Generally Synchronous Motors can be classified in DC excited and non-excited motors. The first configuration requires a direct current supply for the rotor, that presents physical windings throw which flow the excitation current that is responsible of generating the rotating magnetic field. Due to the physical windings the DC excited motors are generally larger in size and consequently has more current losses. Not-excited motors presents no windings; consequently, the Joule losses are reduced.

[4] //it.wikipedia.org/w/index.php?title=Motore_sincrono&oldid=127654252

The following image show the equivalent circuit of the generic synchronous machine in both configurations: motor and generator.

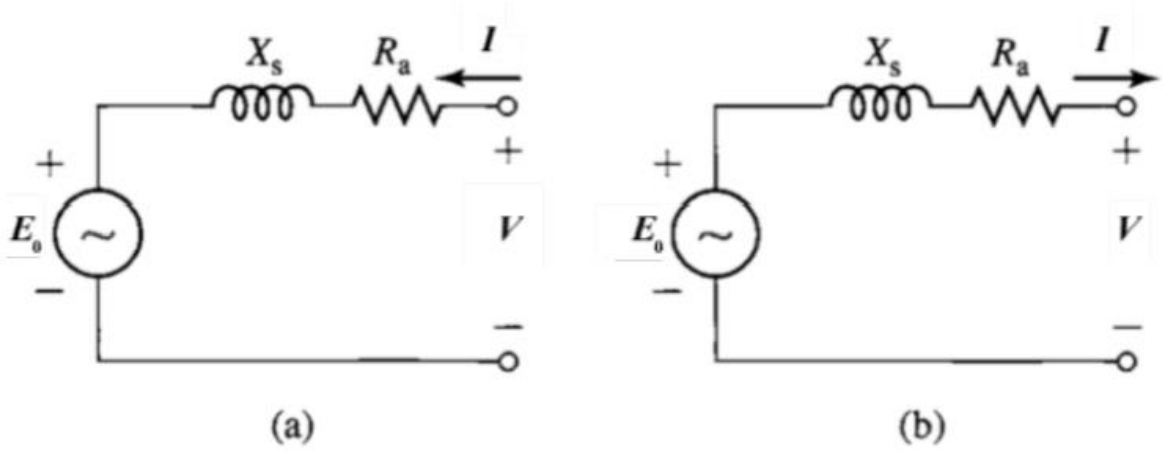

Figure 1.8 SM equivalent circuits: a) Motor reference direction. b) Generator reference direction.

From the equivalent circuit of the generic synchronous machine, it's possible to write the following equation. The synchronous reactance X_s is obtained by the sum of the equivalent stator and rotor reactance, this is of course an approximation because the phenomenon is non-linear, but it makes easier to evaluate the working principle. Appling the Kirchhoff law in the generator configuration:

$$\boldsymbol{V = E_0 - (R_a + JX_s)I}$$

Depending on the size of the Machine the contribution of the armature resistance R_a can be negligible with respect to the other one. In Figure 1.8 it is not considered for the vector representation. In Figure 1.9 are also presented these parameters:

- ***δ load angle,*** important for defining the operating mode of a Synchronous Motor;
- ***φ phase angle***, between the load voltage and the corresponding current;
- ***β current angle***.

$$\overline{CB} = X_s I\, cos(\varphi)$$

$$\overline{CB} = E_0 \sin(\delta)$$

$$X_s I \cos(\varphi) = E_0 \sin(\delta)$$

In this condition, the generated active electric power is:

$$P_e = 3VI\,cos(\varphi) = \frac{3VE_0}{X_s} sin(\delta)$$

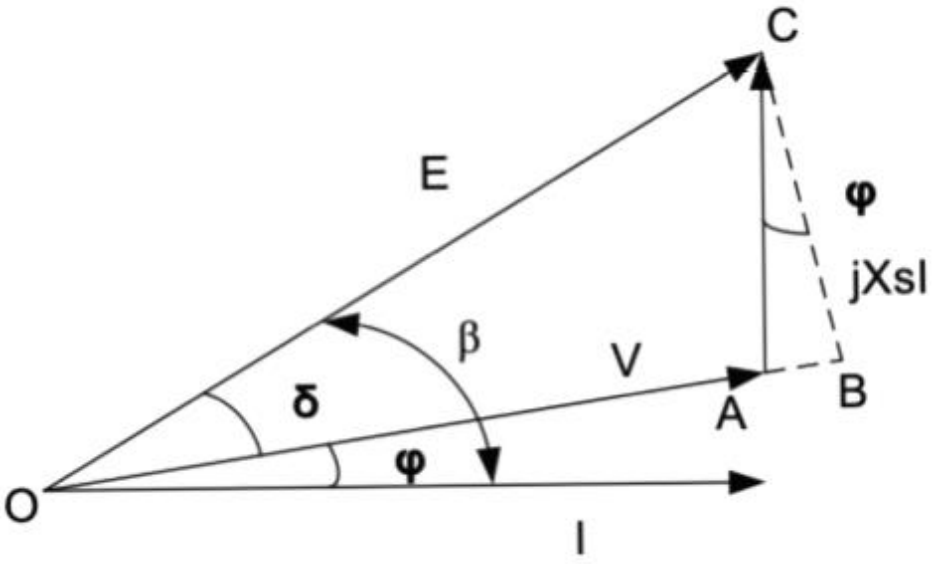

Figure 1.9 SM vector diagram Voltage a Current representation.

As it is shown a factor 3 is involved in the equation due to the three-phase nature of the system, another factor involved is the cos(φ) also known as "power factor" which represents the ratio between the real power and the apparent one flowing in the circuit. The corresponding motor torque is obtained by equalizing the mechanical and electric power. In this case, mechanically no efficient assumptions have been done, so ideal conditions are assumed. This is not a strong approximation because mechanically the wasted part of energy is very low in this kind of system being present no mechanical friction.

$$P_e = P_M = Tw_m$$

$$T = \frac{3VE_0}{\omega_m X_s} sin(\delta)$$

As mentioned, the mechanical speed ω_m is directly related with the electric pulsation ω_e remembering the relationship that involves the number of pole pairs pp ($\omega_e = \omega_m pp$):

$$T = pp\frac{3VE_0}{\omega_e X_s} sin(\delta)$$

The following Figure 1.10 shows the trend of the generated Torque with both motor and generator mode highlighted: the positive sign of the generator mode is basically due to a convention that underlines the fact that the Torque is the product or the cause of the rotating magnetic field. Taking into consideration only one field of working (Figure 1.11) is clear that 2 different fields of stable and unstable condition can be individualized. To shortly remember what a stable region is it is possible to say: a stable region is characterized by corresponding growth of both load angle and Torque.

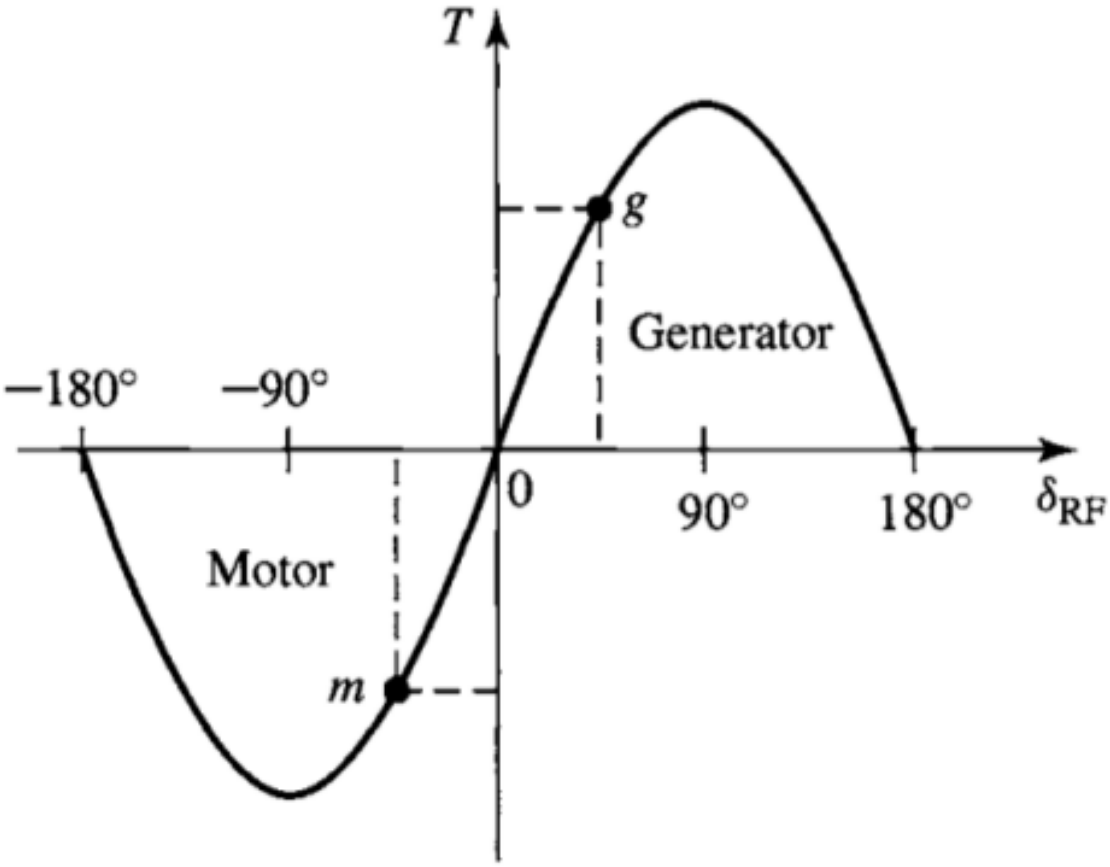

Figure 1.10 Synchronous machine torque - load angle characteristic.[5]

[5] Fitzgerald, A., Kingsley, C., & Umans, S. (2003), Electric Machinery (Sixth Edition), New York, New York, United States: McGraw-Hill. op. cit., p. 247.

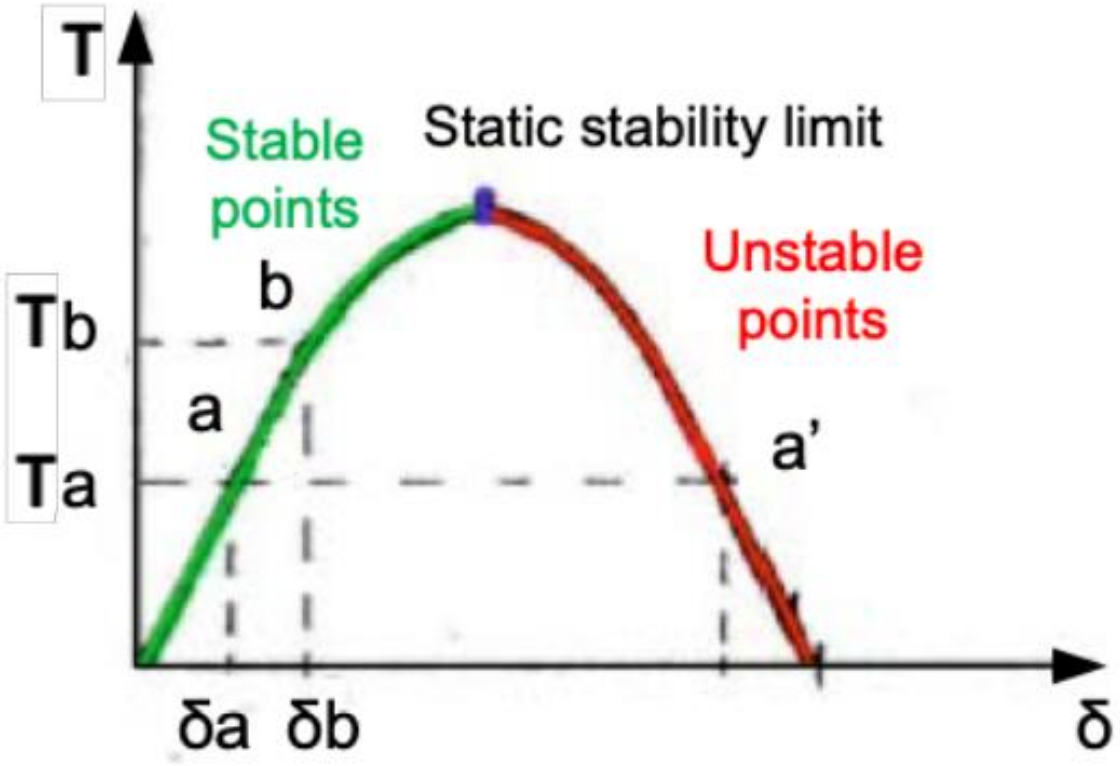

Figure 1.11 Synchronous machine torque - load angle characteristic, zoom on the generator region for showing stable and unstable working points.[6]

For this activity the synchronous electric motor used is an IPMSM (Internal Permanent Magnets Synchronous Motor) where "internal" means that the 5 pole pairs are drowned directly inside the rotor structure, another possible configuration known is the SPMSM (Surface Mounted) where the pole pairs are mounted on the surface of the rotor. The main difference between the two configuration is related to the saliency property of the rotor; as is shown in Figure 1.12 and Figure 1.13 since the rotor is made of materials with a very low magnetic permeability (close to 0) the air gap increase a lot in the direct axes and this create an important anisotropy in the magnet field. Of course, this relative difference between the two magnetic field determinates the same difference in terms of the resulting magnetic flux in the two axes. This became possible only recently with the usage of new rare-earth elements as neodymium-iron-boron magnets.

[6] Rossi, R (2018-2019). Design of a PMSM Field-Oriented Control Algorithm. op. cit., p. 60.

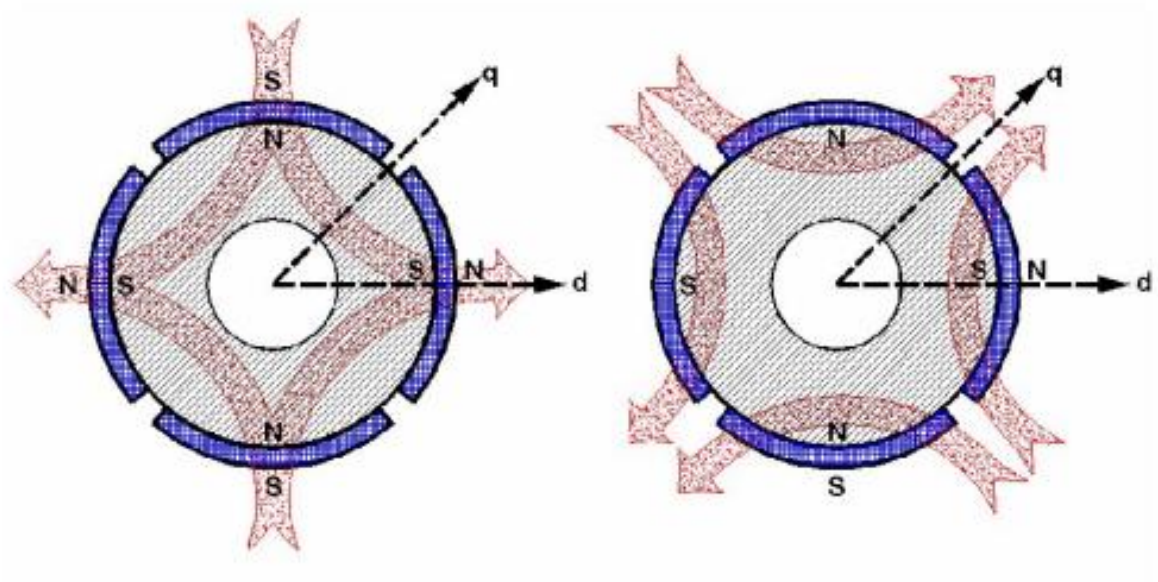

Figure 1.12 Direction of the magnetic flow of a Surface Mounted Permanent Magnet Synchronous Motor (SPMSM).

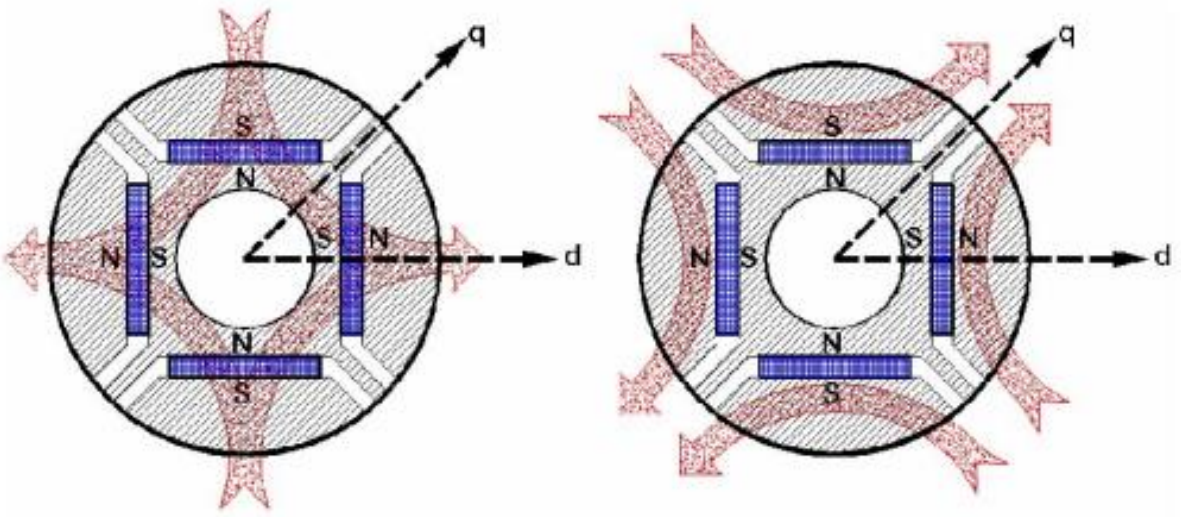

Figure 1.13 Direction of the magnetic flow of an Internal Mounted Permanent Magnet Synchronous Motor (IPMSM).[7]

The dynamic equations known in literature clearly show that for Internal magnets is possible to distinguish 2 contributions to the electrical torque, one is due to the synchronism (so also present for

[7] L.Salvatore – Motore sincrono a magneti permanenti – Politecnico di Bari

surface mounted magnets), the second one is directly due to the high saliency: in fact, the relative difference between the 2 magnetic field creates a “reluctance” torque contribution.

In Figure 14 is possible to notice the two contributions and their resulting electric torque for each value of the angle β between the two axes. The term called “Field Torque is of course the synchronous one and is easy to notice that is always positive and a constant in the working area between the 2 extremes (β = 0 when the two axes are coincidental, and β = 180 when they are in opposition). Also, the graph suggests easily which will be the best value of the angle β to choose for the working conditions, and this is where the “Total Torque” reach his maximum, around 135 degrees, so in a d-q reference frame graphic is easy to understand that the working points will stand in the II square with negative value of direct contribution and positive quadrature one. Is obviously that if the motor turn itself into a generator the quadrature contribution become negative, and the working square become the III.

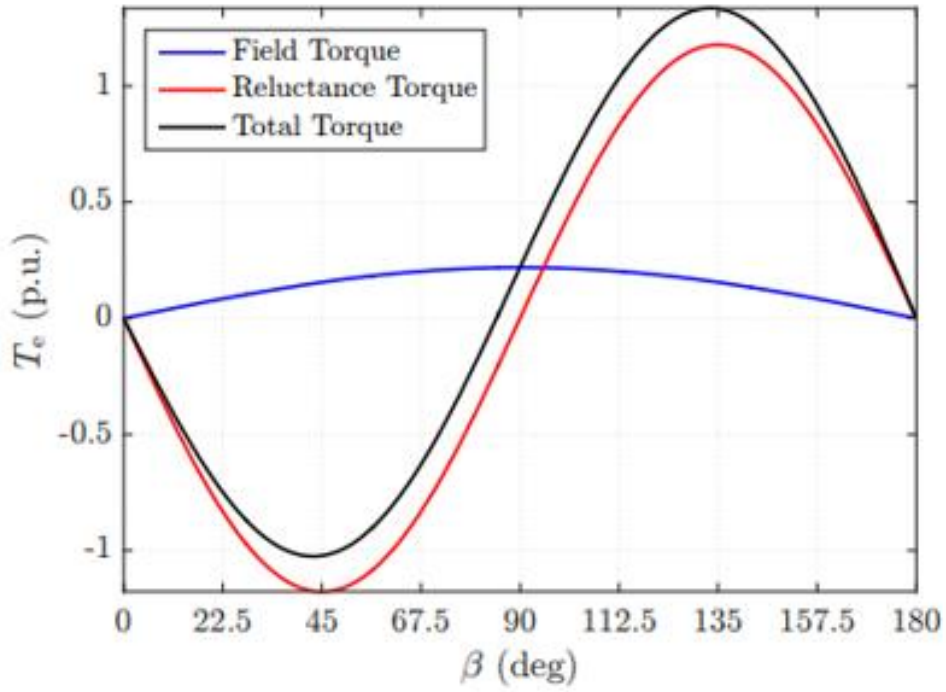

Figure 14 Electromagnetic torque trend for each value of the angle β between the 2 d-q axes. There are shown the two contribution of Torque reluctance and synchronism and their resulting torque.

The sinusoidal currents flow in the stator winding generating the rotating magnetic field make the rotor rotate at the same rate, so, in the case of an anisotropic machine an additional reluctance torque

is added. Since, as shown before this depends on a pulsation that is 2 times higher in frequency the term will depend on the sin (2δ).

$$T = pp\frac{3VE_0}{\omega_e X_s}sin(\delta) + 3pp\frac{V^2}{2\omega_e}\left(\frac{1}{X_q} - \frac{1}{X_d}\right)\sin(2\delta)$$

where X_q and X_d are the equivalent quadrature and direct reactance respectively, expressed in the two-phase rotating reference frame synchronized with the rotor (obtained from the Clarke-Park transformations).

Permanent Magnets Synchronous Motor (PMSM)

A Permanent Magnet Synchronous Motor uses permanent magnets embedded to the steel to produce the magnetic field, when the AC supply connected with the stator trough the windings provide the currents the makes the magnetic field rotate and tend to the same speed the AC impose with the frequency it works (so the synchronism is guaranteed). PMSMs has so many different applications due to their affordability and high performance in terms of Torque and Power density. In terms of cost there are quite expansive because of the material required for the embedded magnets (rare-earth elements such as Neodymium the guarantees the best performance) has only recently been opened to the market so their production prize is very high, by the way nowadays ferrite magnets have been employed in IPM and SPM motors for reducing the economic impacts. They are commonly used for Elevators, and all those applications where a strong affordability is required, but the biggest market is recently the Automotive sector.

In Electric Vehicles and Hybrid-Electric Vehicles both configurations are needed, the motor of course is used in the powertrain system, but also the generator is essentially for refilling the batteries by exploiting the regenerative braking system or the downhill driving: the kinetic energy of the rotating shaft is converted into electric energy, so opposite-sign currents are generated in the stator and the energy source can be recharged. As it has been discussed in the previous paragraph the absence of physic currents in the rotor (low losses and low inertia) makes them convenient also in term of size ratio and. Indeed, for a Vehicle it's easy to understand that the total weight is a very parameter to look at, for many different reasons. Also, has it's mentioned in the last paragraph the starting issue is overcome using an Inverter that makes possible the speed variation and some different control strategies that will be discussed in the following sections. Concerning the field weakening region

(that will be discuss in detail in section 3.2.5.) that allows the machine to make the speed overcame is rated value.

The Figure 1.14 shows the equivalent circuit of PMSM. This is useful to see the graphically the inter-dependency between the two currents and inductance.

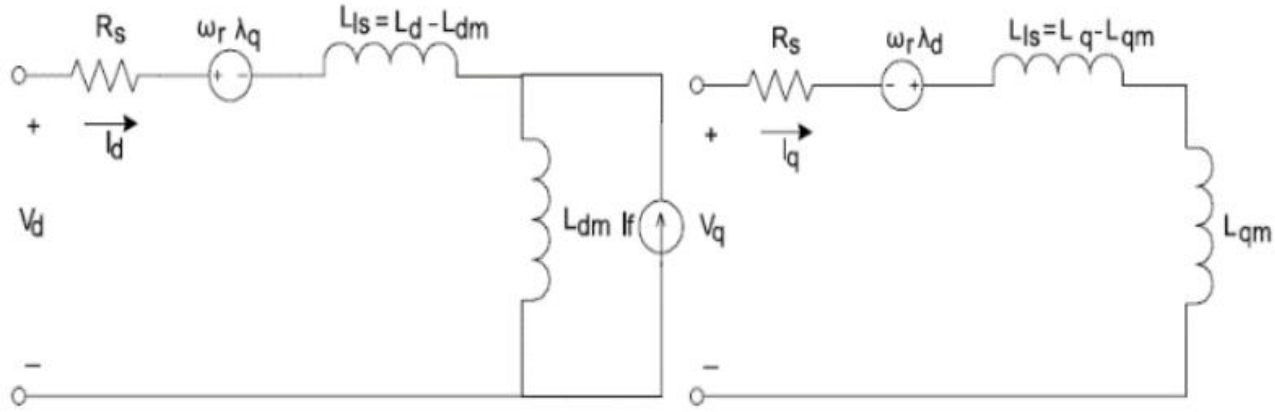

Figure 1.14 Direct-Quadrature equivalent circuit of PMSM.[8]

Internal Permanent Magnets Synchronous Motor (IPMSM)

During the design phase of a PMSM, two possible solutions are generally used for implementing the permanent magnets in the rotor: surface-mounted permanent magnets or interior permanent magnets. In surface-mounted configuration, the permanent magnets are mounted on the outer periphery of motor lamination: this structural arrangement causes a lower integrity and a less robust mechanical action. The principal aspect is the absence of anisotropy and the corresponding null reluctance torque. This system, in fact, can act using only the torque due to the flux linkage between the rotor permanent magnet field and the stator electro-magnetic field. This kind of machines is considered isotropic because the airgap is uniform, so the properties of the magnetic field distribution in the airgap are the same in all directions. In the maximum torque per ampere (MTPA) region – in constant torque condition – the value of the maximum deliverable torque is reduced with an equivalent interior permanent magnet motor, and the d-axis stator current component is imposed equal to zero – its module is increased only in constant power region. Moreover, SPM motors are more efficient in low-

[8] Amin, F., et al. (2017, May), Modelling and Simulation of Field Oriented Control based Permanent Magnet Synchronous Motor Drive System, In Indonesian Journal of Electrical Engineering and Computer Science, 6(2), op. cit. p. 4.

speed range, because the absence of anisotropy and the reduced magnetic field make smaller the field weakening area.[9] In fact, if the inductances are very low – such as in SPM servo motors – the field weakening doesn't provide substantial advantages, because the possibility of increasing the speed over than the rated value is limited. As described in next paragraph, in a SPM configuration the reluctance variation between the direct and quadrature axes inductances are small because these magnets have very low permeability and can be regarded as air in inductance calculations; the SPM motor has very low inductance saliency. The control strategy for this configuration is obviously easier due to the simpler structure.

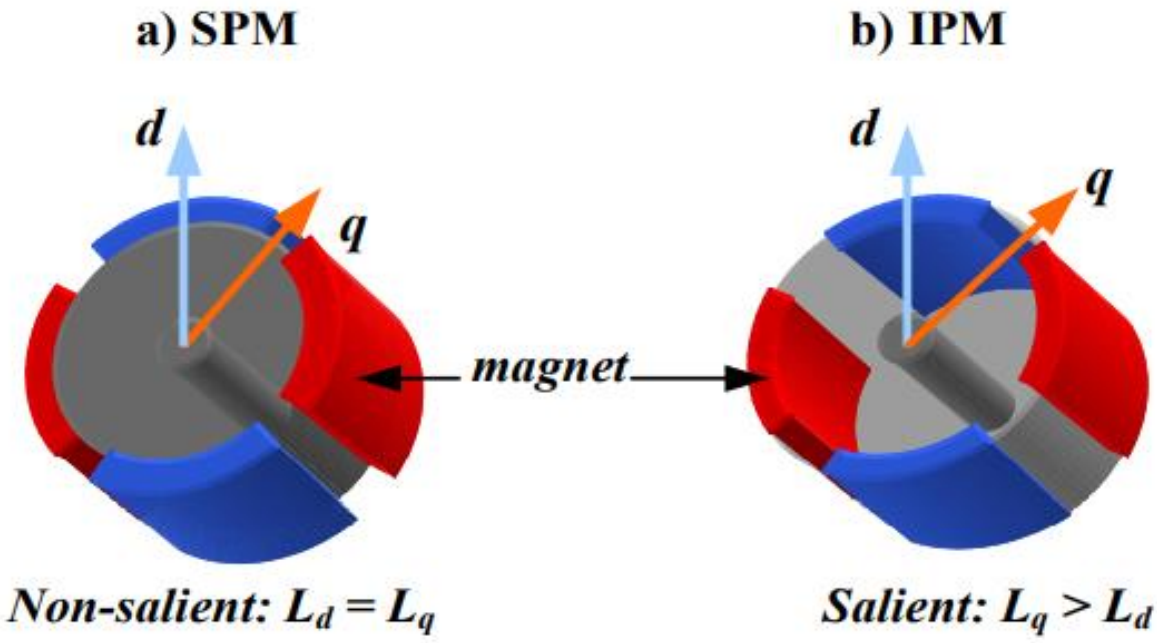

Figure 1.15 Schematic of surface-mounted (a) and interior permanent magnet (b) synchronous machine.[10]

For decades, surface-mounted PMSMs dominate the market; in recent years, due to the emerging hybrid-electric and electric vehicles, the diffusion of IPM solutions is being boosted, because they can guarantee better performance in terms of traction and speed-range. Interior permanent magnet synchronous motors are more performing in terms of torque density (and consequently power density) because they exploit the anisotropic properties of the airgap. The magnets are embedded within the

[9] Krishnan, R. (2010), Permanent Magnet Synchronous and Brushless DC Motor Drives, Boca Raton, Florida, United States: CRC Press, p 34.

[10] Chin, Y., & Soulard, J. (2003), A Permanent Magnet Synchronous Motor for Traction Applications of Electric Vehicles, Presented at IEEE Electric Machines and Drives Conference, Madison, Wisconsin, United States, p. 1

rotor, and they produce the constant rotor magnetic field; since the magnets have lower permeability than iron, the effective air-gap in the magnetic flux path varies depending on the rotor position. The magnetic saliency results in variation of the inductance at the motor terminal with the rotor position – rotor position and measured inductance are always related. These motors have a more intensive magnetic field in the airgap with respect to previous typology; moreover, the magnets can be thinner than those of surface-mounted PMSM for delivering the same torque. In this configuration, an identical SPM power output can be achieved with higher efficiency and smaller stator excitation. In fact, the resulting torque is the addition of isotropic and anisotropic (or reluctance) torque contributions, producing a higher maximum value than the corresponding value in a SPMSM. In the MTPA region, in this case, also the contribution of the d-axis current is fundamental for producing the additional reluctance torque. Thanks to saliency and anisotropic properties, the field weakening region can be extended and the speed-range becomes wider: this is a fundamental aspect that motivates the diffusion and adoption of IPM synchronous motors in HEV/EV applications.

Interior PMSM Dynamic Equations

In this paragraph, the dynamic equations of an interior permanent magnet synchronous machine are described. The IPM configuration is considered, because it is the most general one; for SPM, the necessary modifications in the equations will be described at the end of this section.

Firstly, is useful to clarify that the flux is directly depended (linearity) by the corresponding currents, but this is not true in general. With respect to Figure 1.16 is clear that of course iron saturation matters a lot in the reality. So, the approximation used in this document will be accurate only for a range of currents value that will be imitated by the magnetic properties of the materials involved in the design.

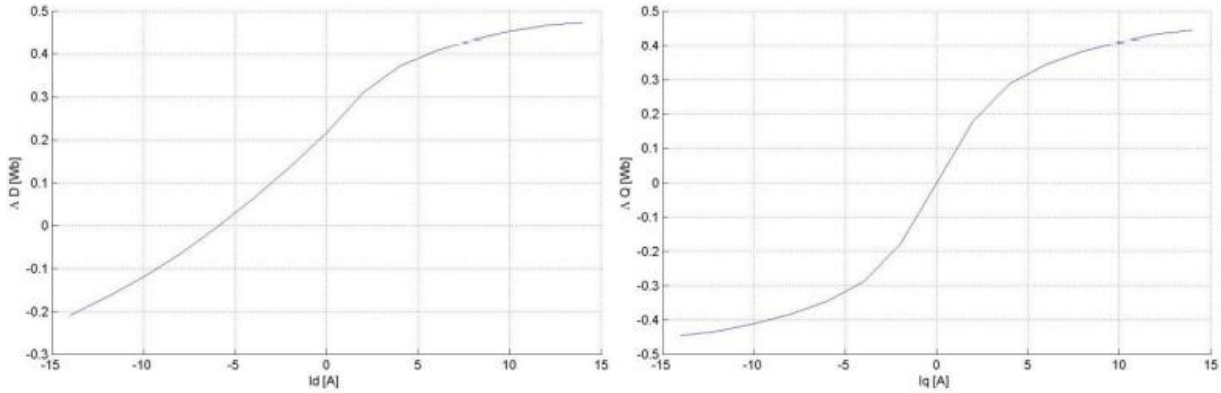

Figure 1.16 Magnetic Flux trend of λ_d and λ_q with current variation.[11]

[11] Vezzaro M.(2009/2010) Caratterizzazione sperimentale di un motore sincrono a magneti permanenti mediante la misura dei flussi in presenza si saturazione incrociata, Università degli studi di Padova, Italy, op. cit. p. 15.

The Clarke-Park transformations are introduced and applied to convert a three-phase system to an equivalent rotating two-phase system: thanks to this mathematical operation, the vector control or field-oriented control (FOC) can be exploited for managing the PMSM in a more efficient and more performing manner. The effective FOC implementation will be discussed in Chapter 3.
Starting from the dynamic system of a generic DC-excited synchronous machine, the equations can be simplified removing the term of the excitation current and using the flux linkage contribution due to permanent magnets. In a PMSM, there is no i_f the stator, so the terms λ_{ra}, λ_{rb}, λ_{rc} are added – they represent the flux linkages established in the stator abc phase windings respectively, due to the presence of PMs on the rotor. These values change depending on the electric angle θ_e: of the electric quantities.[12] Also mutual inductances are considered symmetrical, so M_{ij}= M_{ji} , the core saturation and winding leakage inductance are neglected, the magnetic potential in the airgaps considered sinusoidal, and the higher harmonic waves in the magnetic field are ignored.

So, it's now possible to write the 3-phase flux value as:

$$\begin{cases} \lambda_a = L_a i_a + M_{ab} i_b + M_{ac} i_c + \lambda_{ra} \\ \lambda_b = L_b i_b + M_{ba} i_a + M_{bc} i_c + \lambda_{rb} \\ \lambda_c = L_c i_c + M_{ca} i_a + M_{cb} i_b + \lambda_{rc} \end{cases}$$

Where:

$$\begin{cases} \lambda_{ra} = \lambda_{PM} \cos(\theta_e) \\ \lambda_{rb} = \lambda_{PM} \cos(\theta_e - 120°) \\ \lambda_{rc} = \lambda_{PM} \cos(\theta_e + 120°) \end{cases}$$

The term λ_{PM} is the nominal flux linkage due to rotor permanent magnets. Then:

$$\begin{cases} v_a = r_a i_a + \dfrac{\mathrm{d}\lambda_a}{\mathrm{dt}} \\ v_b = r_b i_b + \dfrac{\mathrm{d}\lambda_b}{\mathrm{dt}} \\ v_c = r_c i_c + \dfrac{\mathrm{d}\lambda_c}{\mathrm{dt}} \end{cases}$$

[12] Ohm, D. (2000), Drivetech Inc., Retrieved from Dynamic Model of a PM Synchronous Motor. op. cit. p. 1.

Those are the voltages 3-phase in the phase plane, with the Clarke transformation is possible to pass into a two-phase α-β reference frame where if the abc reference frame is composed of three axes, displaced each other by 120°, the α-β's one will be composed of two perpendicular axes. Considering that the magnitude of the electric quantities must be conserved, a 3/2 multiplication factor is inserted into the final torque equation as if it should represent the rms value.

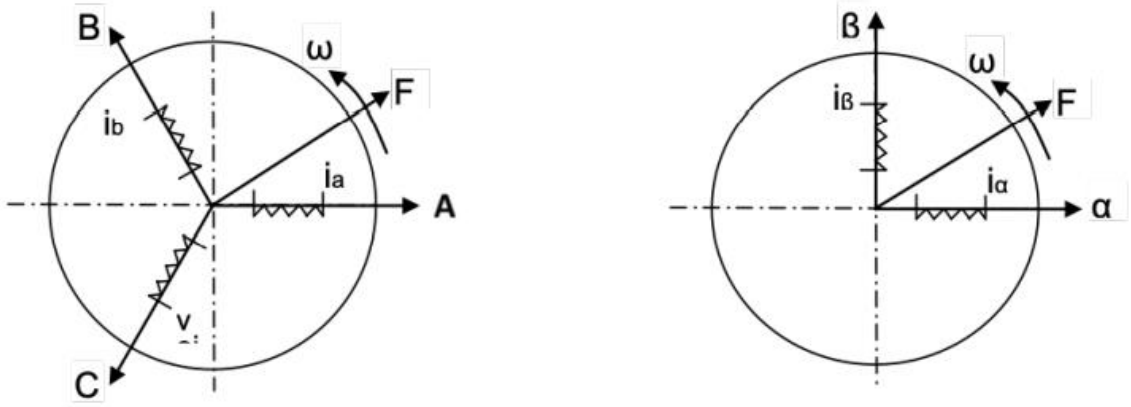

Figure 1.17 Three-phase balanced AC currents on the left, two-phase balanced AC currents on the right.[13]

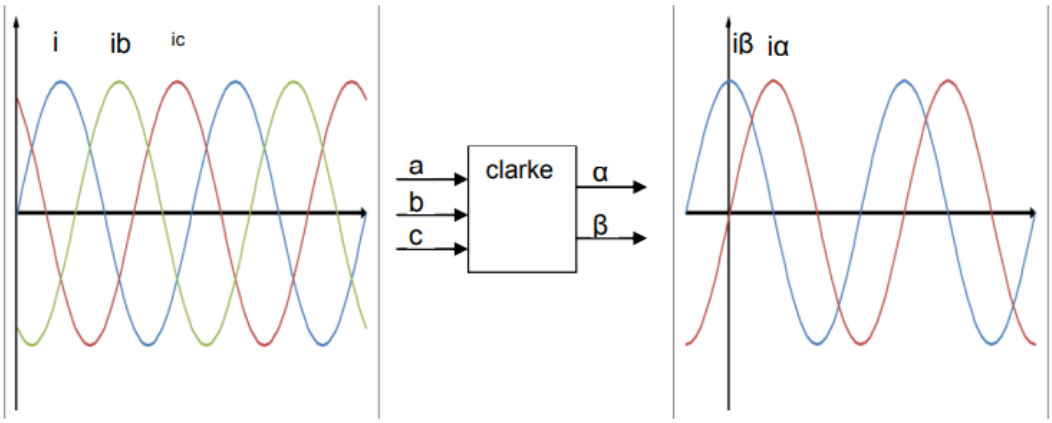

Figure 1.18 Current waveforms with Clarke transformation.[14]

[13] Cypress Semiconductor (2017), Retrieved from Coordinate Transform in Motor Control: https://www.cypress.com/file/222111/download, p. 2.

[14] Cypress Semiconductor (2017), op. cit., p. 4.

$$\begin{cases} x_a = X\,sin\,(\omega t) \\ x_b = X\,sin\,(\omega t + 120°) \\ x_c = X\,sin\,(\theta_e - 120°) \end{cases}$$

$$\begin{cases} x_\alpha = x_a = X\,sin\,(\omega t) \\ x_\beta = \dfrac{\sqrt{3}x_a}{3} + \dfrac{2\sqrt{3}x_b}{3} = X\,sin\,(\omega t + 90°) \end{cases}$$

Here there is also presented the equations useful to do the inverse operation (Clarke^{-1} transformation)

$$\begin{cases} x_a = x_\alpha \\ x_b = -\dfrac{1}{2}x_\alpha + \dfrac{\sqrt{3}}{2}x_\beta \\ x_c = -\dfrac{1}{2}x_\alpha - \dfrac{\sqrt{3}}{2}x_\beta \end{cases}$$

Similarly, the Park transformation is used to obtain a rotating dq reference system. In this case, the measured angle θ of the rotating axes is required. It is fundamental to notice that the resulting generic quantities x_d and x_q are DC quantities so it's possible to apply the Field-Oriented-Control (Section 1.5) this because as it's shown in Figure 1.20 they are no more sinusoidal values and this let to exploit to decouple the stator current components in a PMSM as you could do in a separated excitation DC motor and for regulating the motor in a more efficient manner through the FOC.

$$\begin{cases} x_d = x_\alpha \cos(\theta) + x_\beta \sin(\theta) \\ x_q = -x_\alpha \sin(\theta) + x_\beta \cos(\theta) \end{cases}$$

As usual, also the Park inverse transformation is possible (Park^{-1}):

$$\begin{cases} x_\alpha = x_d \cos(\theta) - x_q \sin(\theta) \\ x_\beta = x_d \sin(\theta) + x_q \cos(\theta) \end{cases}$$

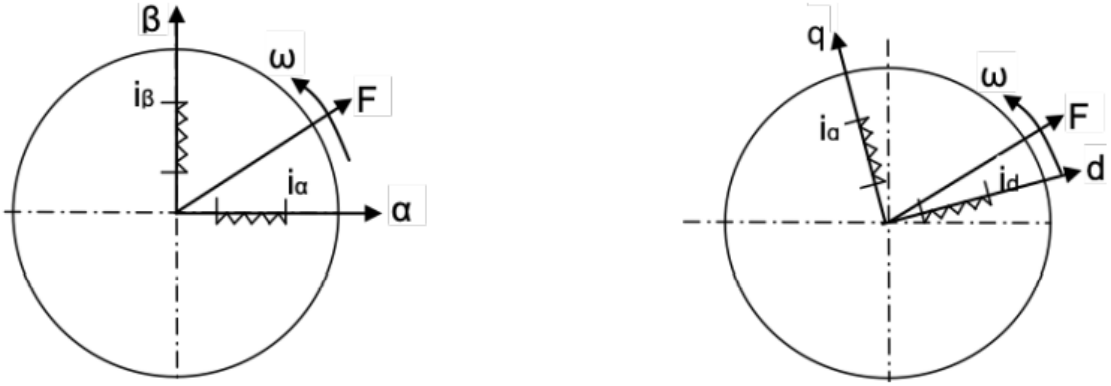

Figure 1.19 Currents in αβ reference frame on the left, current in rotating dq reference frame on the right.[15]

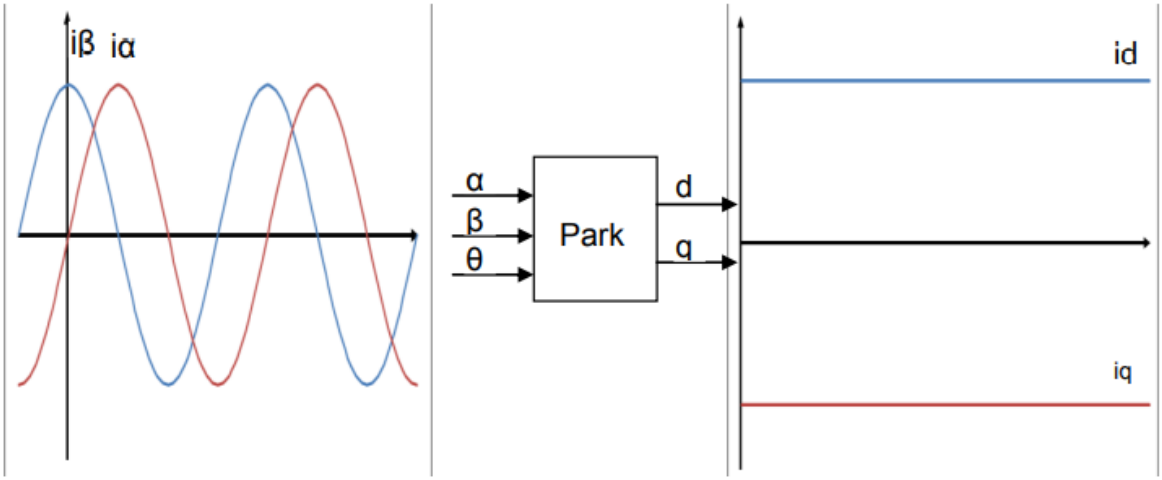

Figure 1.20 Current waveforms with Park transformation.[16]

For the PMSM, the used angular position for the mathematical operation is the electrical angle θ_e: because in the AC machine's stator there are electric quantities. In fact, Clarke-Park transforms are used for obtaining the representations of stator currents, termination voltages and flux linkages in the rotor d-q reference frame. From the field-oriented control block scheme of successive chapter it is possible to see that sensors are used for measuring the rotor mechanical position; this value is then multiplied for the number of pole pairs to obtain the rotor electrical angle. In a similar manner, the corresponding electrical speed is computed from the measured mechanical velocity of the rotor. With

[15] Cypress Semiconductor (2017), op. cit., p. 6.

[16] Cypress Semiconductor (2017), op. cit., p. 7.

this approach, the Clarke-Park direct and invers transformations are correctly implemented, and stator voltages and currents maintain the correct pulsation.[17] By using this strategy for simplifying the analysis and the control of a PMSM, the obtained stator quantities in the rotor d-q-axes are the following ones. All the values are referred to the rotor RF, so the apex r is omitted. In the chosen configuration, the rotor d-axis is aligned along the PMs' magnetic flux direction.

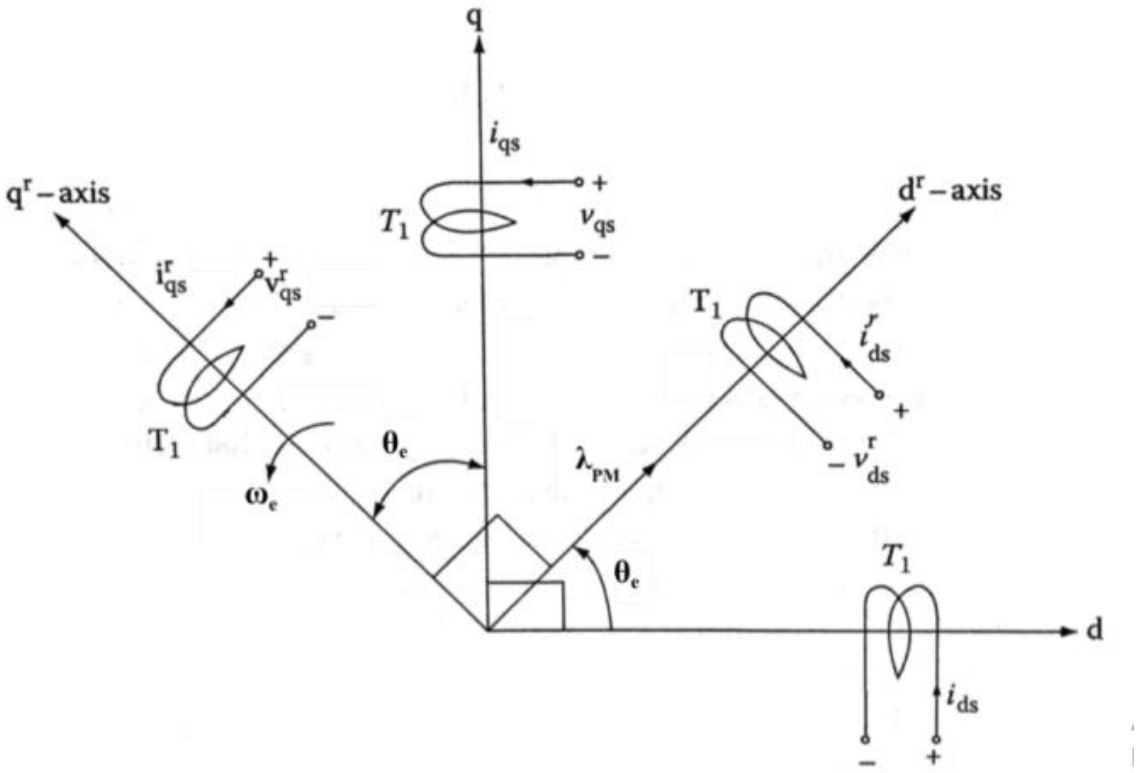

Figure 1.21 Stationery and rotor d-q reference frames.[18]

The variables that appear in the equations are:

- λ_d, λ_q stator magnetic flux linkages in the rotor d-q reference frame;
- λ_{PM} armature flux linkage due to rotor permanent magnets;
- $i_{d,}$ i_q stator currents expressed in the rotor dq RF;
- $v_{d,}$ v_q stator voltages expressed in the rotor dq RF;
- $L_{d,}$ L_q direct and quadrature axes stator inductances in the rotor d-q RF, with $L_d > L_q$;

[17] (Rossi, 2018-2019), op. cit. p. 70.

[18] Krishnan, R. (2010), op. cit., p. 232.

- R_s stator resistance,
- ω_e rotor electrical speed;
- ω_m rotor mechanical speed;
- pp number of pole pairs;
- T_{em} electromagnetic torque (motor torque), composed of a synchronous contribution $T_{em,syn}$ and a reluctance contribution $T_{em,rel}$ in a generic IPMSM;
- T_r resistance torque;
- J motor moment of inertia;
- K_T torque constant.

$$\begin{cases} \lambda_d = L_d\, i_d + \lambda_{PM} \\ \lambda_q = L_q i_q \end{cases}$$

So finally, we can express the correct relations between current and the voltages directly in the d-q reference frame. These equations are useful in the FOC technique because they will be directly implemented inside the current PI control to obtain the corresponding V_d and V_q quantities.

$$\begin{cases} v_d = R_s\, i_d + \dfrac{d\lambda_d}{dt} - \omega_e\, \lambda_q = R_s\, i_d + L_d \dfrac{di_d}{dt} - \omega_e\, L_q\, i_q \\ v_q = R_s\, i_q + \dfrac{d\lambda_q}{dt} - \omega_e\, \lambda_d = R_s\, i_q + L_q \dfrac{di_q}{dt} - \omega_e\, L_d\, i_d + \omega_e\, \lambda_{PM} \end{cases}$$

It's important to notice that in FOC strategy the currents i_d and i_q are exantially to the torque computation because they are directly determinated by the combination of speed and torque internal loops. These reference currents are then passed to the vectoral controller which will execute the PI control of the currents separately by the error computed between the reference value and the feedback read by the sensor in the PMSM. Here are reported the inverse formulation of the i_q and i_d:

$$\begin{cases} i_d = \dfrac{1}{L_d} \int \left(v_d - R_s\, i_d + \omega_e\, L_q\, i_q \right) dt \\ i_q = \dfrac{1}{L_q} \int \left(v_q - R_s\, i_q - \omega_e\, L_d\, i_d - \omega_e\, \lambda_{PM} \right) dt \end{cases}$$

What is most important is the equation of the electromagnetic torque T_{em} can be obtained by equalizing the input electric power and the output mechanical power. In a generic interior magnet synchronous motor (where the saliency of the rotor can be exploited for generating additional torque) T_{em} contains a term that is related to q-axis current only and it is called synchronous torque $T_{em,syn.}$ Another term is the $T_{em,rel}$ or reluctance torque which depends on the difference between the equivalent d-axis and q-axis stator inductances. This reluctance torque is added to the other value to increase the torque and power density of these motors, with respect to surface mounted PMSMs:[19]

$$T_{em} = \frac{3}{2} pp\left[\lambda_{PM} + \left(L_d - L_q\right) \cdot i_d\right] i_q = T_{em,syn} + T_{em,rel}$$

$$T_{em,syn} = \frac{3}{2} pp \lambda_{PM} i_q$$

$$T_{em,rel} = \frac{3}{2} pp\left(L_d - L_q\right) \cdot i_d i_q$$

From a mechanical point of view the dynamic equation considers an Inertial factor $J\dot{\omega_m}$, a damping factor $B\omega_m$ and a direct resistence factor (or load) $T_{laod.}$

$$T_{em} = T_{laod} + B\omega_m + J\dot{\omega_m}$$

The sensor measures the angular speed ω_m than integrating it is possible to estimate the angular position θ_m and the corresponding spatial angle θ_e:

$$\begin{cases} \theta_m = \int \omega_m \, dt \\ \theta_e = pp \, \theta_m \end{cases}$$

In a SPMSM, the absence of saliency (d-axis and q-axis reluctance are considered equal and constant) determinates that the motor torque is expressed only as function of i_q, so the synchronous torque

[19] (Rossi, 2018-2019), op. cit. p. 72.

contribute is considered. This relationship depends on motor parameters only on the motor parameters pp and λ_{PM} so the Torque is directly depended on the i_q contribution:

$$T_{em} = T_{em,syn} = Ki_q$$

Where:

$$K = \frac{3}{2}\lambda_{PM}pp$$

As it's been mentioned before the Control Unit purpose is to achieve the MTPA condition, or in other words the combination of d-axis and q-axis currents that determinates the minimum value of current magnitude and achieve the maximum Torque delivery. In Figure 1.22 are shown the two-contribution linked together by a "current angle" β. According with the saliency consideration done is clear that certain values of β determinates condition of more deliverable Torque (more power density), in particular, obviously, when the reluctance contribution is maximum the Torque T_{em} will be maximum too as it is shown in Figure 1.23.

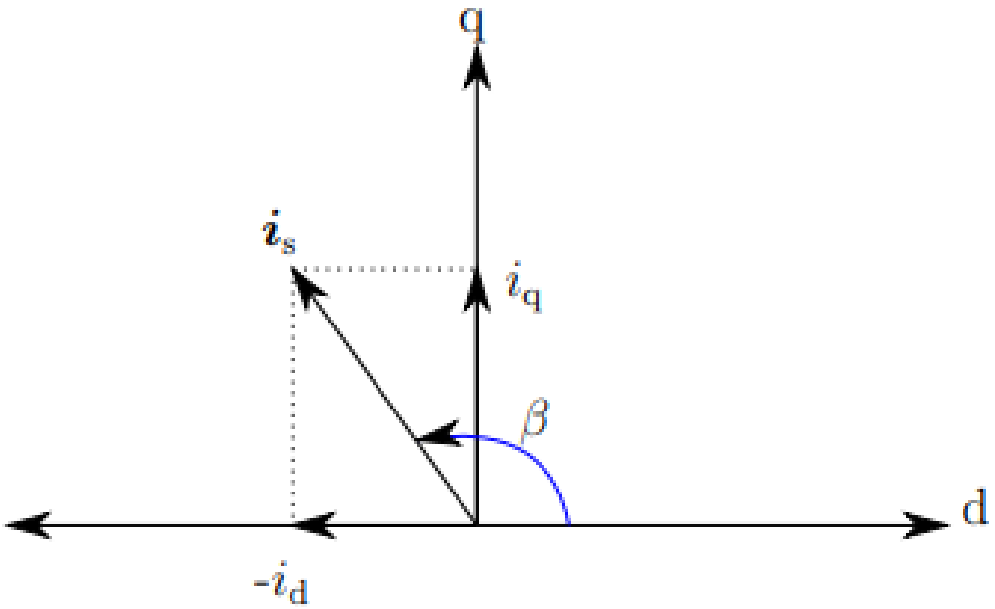

Figure 1.22 Current angle β in the d-q axis reference frame.[20]

[20] Khan W.A. (2016), Torque Maximizing and Flux Weakening Control of Synchronous Machines, Espoo, Finland School of Electrical Engineering. op. cit., p. 11

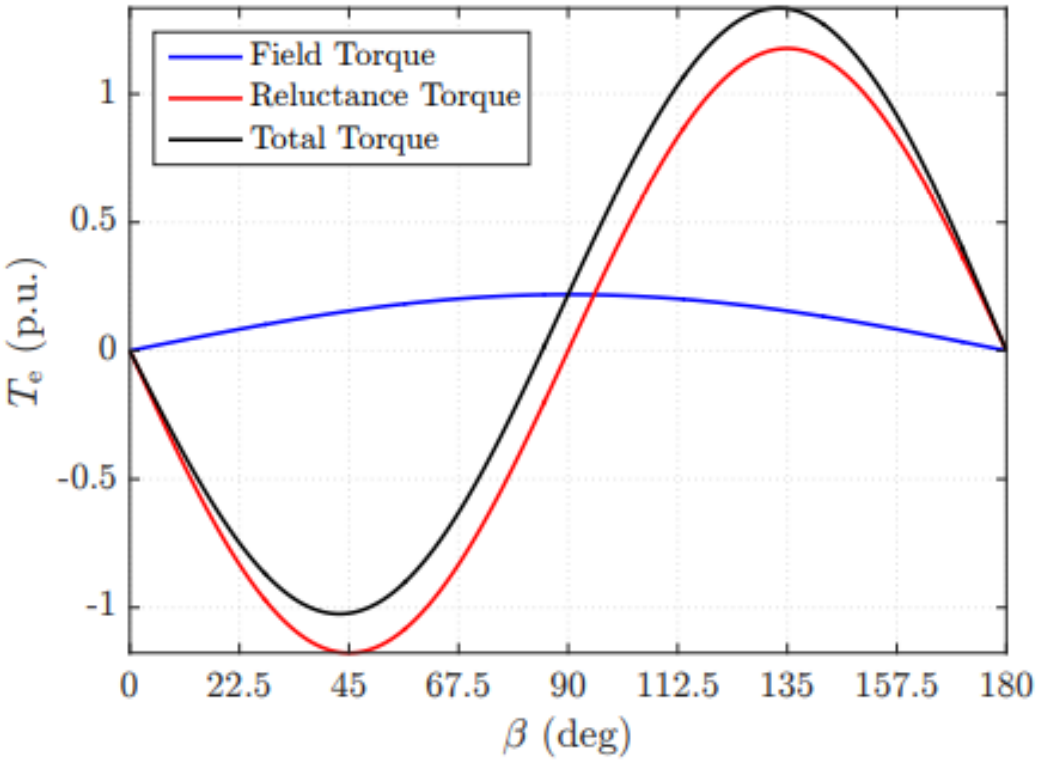

Figure 1.23 Deliverable Torque and current angle[21]

$$\begin{cases} i_d = i_s \cos(\beta) \\ i_q = i_s \sin(\beta) \end{cases}$$

Substituting these relations inside the T_{em} equation above it's obtained:

$$T_{em} = \frac{3}{2} pp \left[\lambda_{pm} i_s \sin(\beta) + \left(L_d - L_q \right) i_s^2 \sin \frac{2\beta}{2} \right]$$

With a simple derivation on the current angle β and equalizing to zero it's possible to get the maximum value of T_{em} for each value of current magnitude value i_s (in Figure 1.24) is represented the trajectory of this curve that is exactly the MTPA curve.

[21] (Khan, 2016), op. cit. p. 12.

$$\frac{\partial T_{em}}{\partial \beta} = \frac{3}{2} pp \left[\lambda_{pm}\, i_s \cos(\beta) + \left(L_d - L_q\right) i_s^2 \cos 2\beta\right] = 0$$

By resubstituting the i_d and i_q as did before we obtain the equation:

$$\left(L_d - L_q\right) i_d^2 + \lambda_{pm} i_d + \left(L_d - L_q\right) i_q^2 = 0$$

Which is no more than a second-degree equation on the variable i_d. To understand wich solution can be physically acceptable it's needed to clarify the domain where the solution is researched. Figure 1.20 shows clearly where Torque gets positive values (Motor conditions). Otherwise, the Regenerative breaking condition will be satisfied for i_q negative values. Figure 1.26 summarize all the information about the MTPA trajectory described till now. The colored hyperbolic curves represent the gradient of MTPA torque values in function of the magnitude i_s . The maximum reachable is tangent to the i_{max} curve it will be introduced in the following paragraph. It's easy to imaginate that for breaking regenerative conditions the Torque hyperbolas will be collocated in the fourth quadrat symmetrical with reference on the i_d axis.

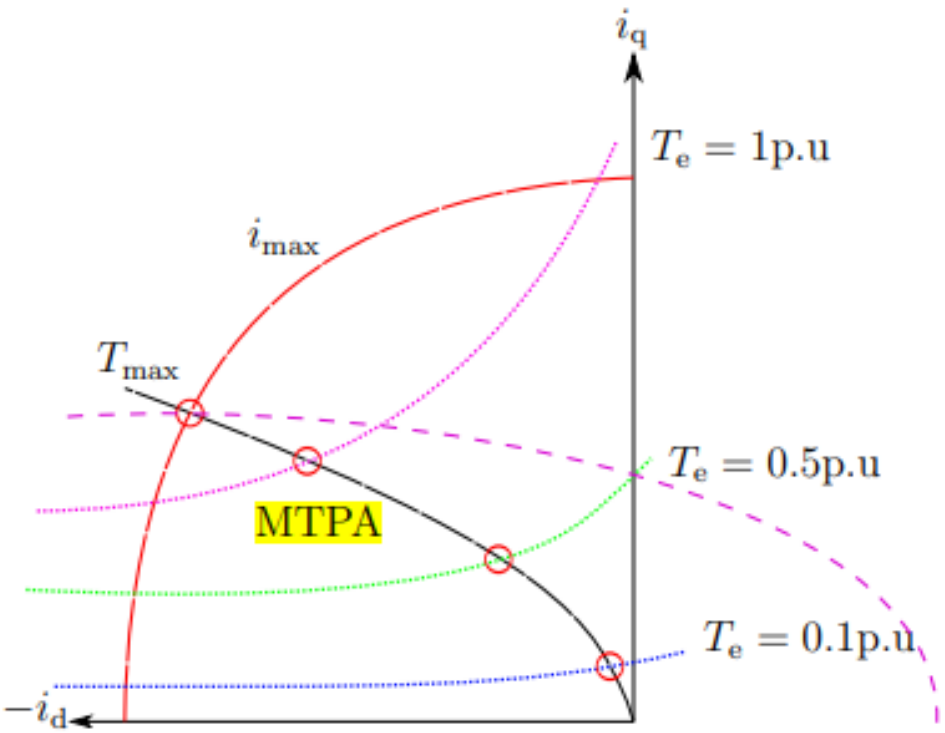

Figure 1.24 Maximum Torque per Ampere Trajectory[22].

[22] (Khan, 2016), op. cit. p. 10.

The only solution acceptable is the following equation:

$$i_{d\_mtpa} = \frac{\lambda_{pm}}{4(L_q - L_d)} - \sqrt{\frac{\lambda_{pm}^2}{16(L_q - L_d)^2} + \frac{i_s^2}{2}}$$

From Figure 1.24 is clear that that Torque achievable increase for bigger magnitude value but of course there is a superior limit which can't be overcome in terms of magnitude this because practical drives system is fed through a power electronic converter that impose a limit on the maximum output voltage and current that the inverter can produce.[23] So practically it's possible to highlight 2 main practical limits:

$$\text{Current limit: } \sqrt{i_q^2 + i_d^2} = i_s \leq i_{max}$$

In the d-q reference frame the current limit constraint takes the form of a circle having a center at origin with a radius of $i_{max.}$

$$\text{Voltage limit: } \frac{\left(i_d + \frac{\lambda_{pm}}{L_d}\right)^2}{L_q^2} + \frac{{i_q}^2}{L_d^2} \leq \left(\frac{v_{s,max}}{\omega_m L_d L_q}\right)^2$$

Where $v_{s,max} = v_{DC}/\sqrt{3}$. Within the d-q plane the voltage limit takes the form of an ellipse having a center at ($-\lambda_{pm}/L_d$, 0). The ellipse encloses all the operating points where the terminal voltage does not exceed the maximum voltage.[24] The Figure 1.25 summarize both the limits in the dq-axis reference frame.

[23] (Khan, 2016), op. cit. p. 10.

[24] (Khan, 2016), op. cit. p. 11.

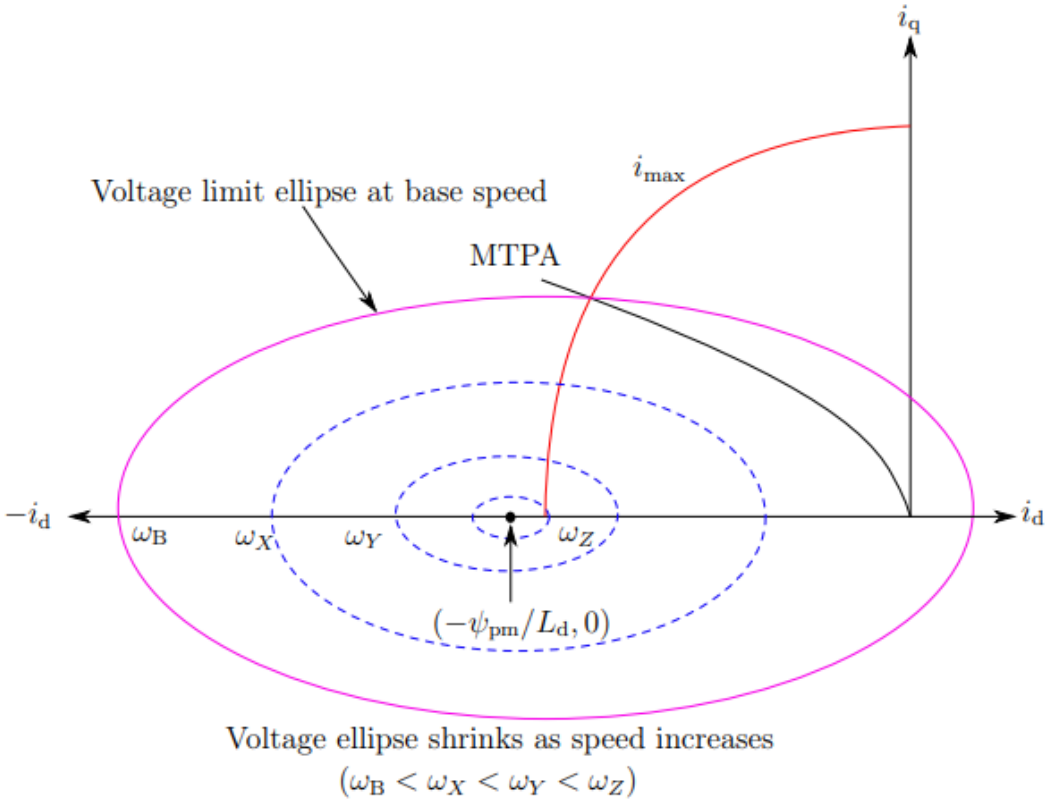

Figure 1.25 Circle diagram exhibiting MTPA trajectory, current and voltage limits.[25]

In Figure 1.25 the MTPA trajectory is represented in black. It's obvious at this point that all the d-q currents couples must be inside the intersection of the 2 figures (The voltage ellipse and the current circle) and if you want to increase the speed this starts to decrease. For these reason Field-Weakening (FW) Control Technique has been implemented: This Technique involves a reduction of the d-axis flux value acting on the air gap and so to the λ_{pm} contribution.

The Figure 1.26 shows how the MTPA trajectory is deflected when the FW control condition is reached. So, the speed continues to increase from A to B and the torque is kept constant (Figure 1.26(a)). It's clear that when point B is reached and the maximum current is obtained the only possibility to continue to let the speed increase is to follow the current circle limit (B-C trajectory). The speed corresponding at the point B conditions (both limits reached) is also known as "base speed" ω_{base} and it represents the maximum value of power reachable from the system (Figure 1.26 (b)) but also the first value of the constant power trajectory.

[25] (Khan, 2016), op. cit. p. 14.

At this point two possibility can be verified:

- The d-axis coordinate of the center of the voltage ellipse has an absolute value bigger than the i_{max} (as it happens in Figure 1.25).
- The d-axis coordinate of the center of the voltage ellipse is inside the i_{max} circle (as it happens in Figure 1.26).

In the first case the speed continues to increase till the i_q value reaches zero along the current circle: the speed continues to increase and the torque to decrease keeping the power constant.

In the second case, how is represented in Figure 1.26, when the point C is reached the trajectory follows the line C-D where the speed can continue to increase till the maximum value (point D) where the ellipse collapse to his center and ideally "infinite" speed value is reached. This last part of the trajectory is called Maximum Torque Per Volt (MTPV) and, similarly to MTPA follow a nonlinear trajectory. This implementation is needed only in peculiar cases of application where infinite drive speed is needed and/or very low torque are needed. This is not the case of an automotive application normally but in any case, the procedure similarly to the MTPA case can be described in the same way by imposing the orthogonality and derivation

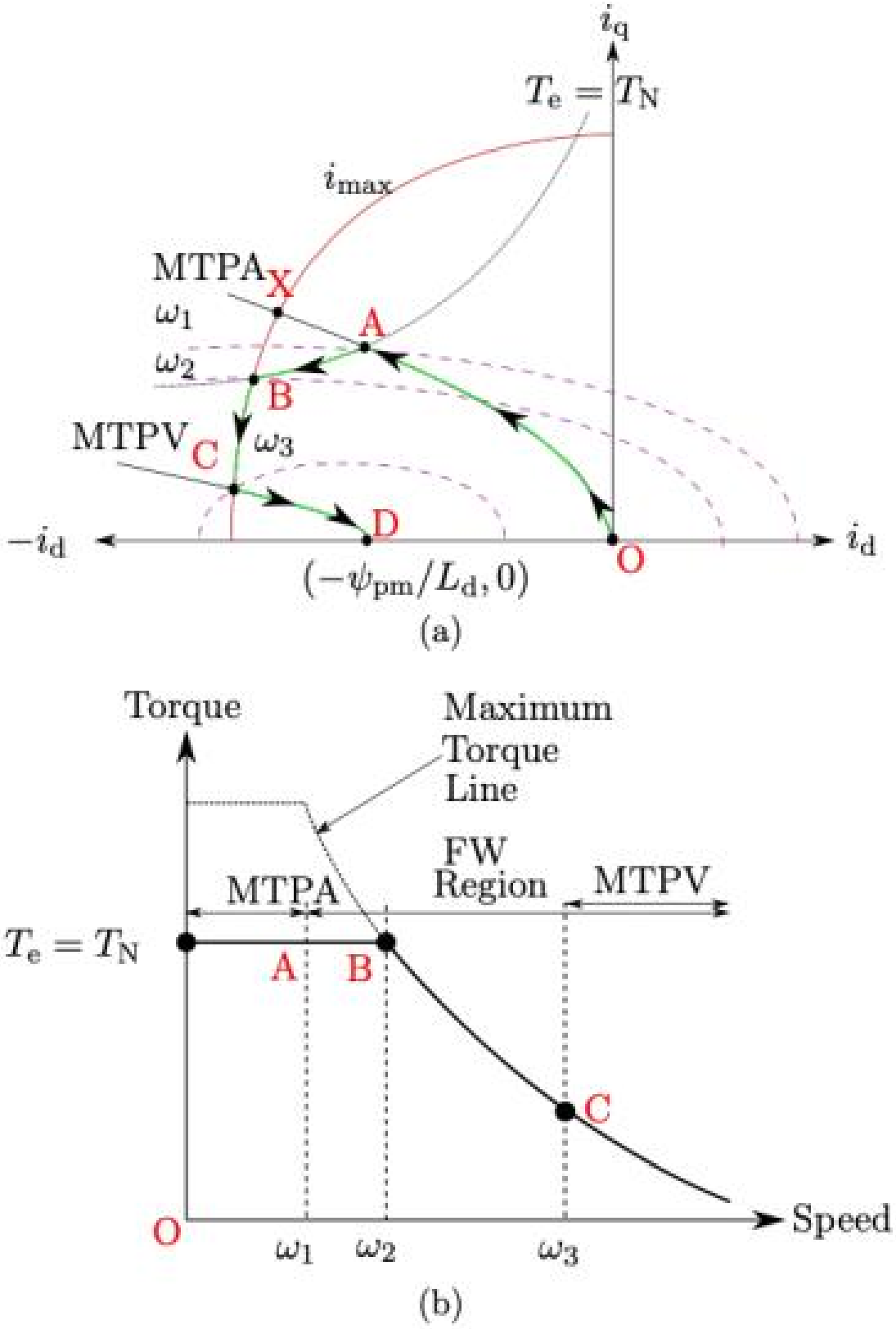

Figure 1.26 a) d-q reference frame with all the conditions in evidence: Nominal MTPA behavior, Flux Weakening and MTPV (Maximum Torque per Voltage). b) Speed-Torque characteristic in evidence: the conditions explicated above.

It's evident that when the center of the ellipse in terms of absolute value is bigger than the i_{max} (as it happens in Figure 1.25) it's not possible to realize the Maximum Torque per Volt condition because the Current limit cannot be reached.

1.4. Voltage Source and Currents Inverter: Modulation Strategies

In general, converters are those kinds of devices whose purpose is to "convert" a form of energy, received as input into another one available as output. Converters can commutate different kind of energy, but when we talk about power converter, we generally refers to those devices which are able to work with electricity. So, basically an inverter is a power converter whose technology is based on the semiconductors. The main purpose of an inverter is the possibility to modulate magnitude and frequency on one/multi phase voltage signals. Those devices had recently become one of the most important segments of the technology market, indeed thanks to the industrialization of the new semiconductors a huge gamma of application has been wide opened across all the technology sector. One of the most required applications is the possibility they opened in the driving of AC motors, which has become of the bigger worldwide market, not only for the automotive sector. In Figure 1.27. is shown a scheme of what can generally describes an AC driving scheme.
As mentioned, the application used is related to the possibility to convert huge quantities of DC currents into the AC form. And this is also the main reason why, at the beginning of the century the electric energy has lost is "battle" against the Internal combustion engines in the automotive sector, simply because the AC current was very expending to be transferred. For the same reason, nowadays the possibility offered by the power converter once again opened the way to the electric propulsion. What an inverter does is to convert a direct voltage/current into an alternating one, that's why we refer to this device as "DC/AC" converter. As said in the field of the converters also "AC/DC", "AC/AC" and "DC/DC" are contemplated. The term "inverter" can be referred to the complete feeding scheme of an AC machine, including the AC/DC converter and the DC/AC inverter.[26] Of course to optimally drive each kind of AC motors a precise gamma of AC currents can be needed in terms of frequency and magnitude.

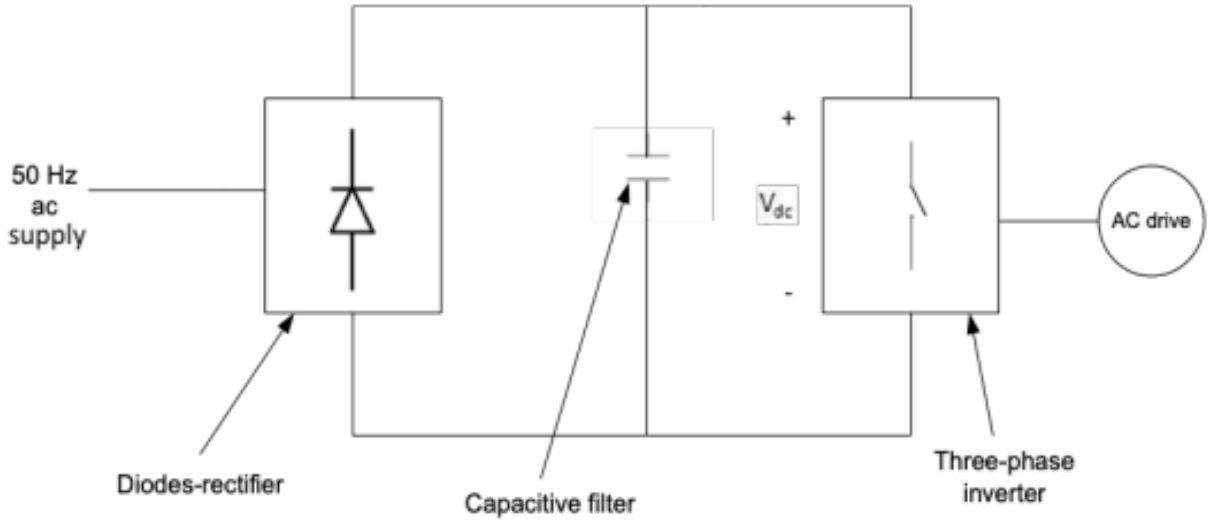

Figure 1.27 Feeding scheme for an AC motor, from the AC electric grid supply.[27]

From Figure 1.27 it's possible to do many observations:

- An electrical grid that supplies the 230 V (50 Hz);
- AC/DC converter is insider in the chain to connect the grid to the DC line;
- A Diodes-rectifier is used to rectify the sinusoidal wave;
- The Capacitive is needed to eliminate the ripples, that's why is named "capacitive filter";

It is also possible that the DC power is provided directly (in this case no AC/DC converter is needed). This is what commonly happens in the electric (BEV) and hybrid-electric vehicles (full Hybrid or MHEV), the power source is typically a battery, and the necessary DC voltage is provided to the inverter directly.

In the case of PHEV (Plug-in Electric Vehicles) for example, also, the battery pack is connected to the external electric grid through a charging system and this system operate as, precedingly described an AC/DC converter. In some application also the regenerative braking energy is exploited for refilling the onboard energy source.
The most common application as induction and synchronous machines needs a 3-phase system which provides the desired three sinusoidal voltages v_a, v_b and v_c where each the waveforms have a phase

[27] Mohan, N., Undeland, T., & Robbins, W. (2003), Elettronica di Potenza. Convertitori e Applicazioni (Terza Edizione), Milano, Italy: Hoepli, p. 223.

difference of 120 degrees respect to the others. In Figure 1.28 a scheme of a 3-phase inverter is reported: it's easily possible to notice three legs (or branches) and 6 switching diodes (two for each branch), the upper and lower switches are used to properly modulate the output reference voltage as desired. This modulation is basically done by turning on and off the switches with a precise timing. Also, commonly in the control application when is needed to control the speed an inverter is always required, the following important relation underlines a strong relationship between the angular speed and the frequency of modulation through the use of number of pole pairs.

$$n = \frac{60f}{pp}$$

Where:

- **n** is the angular velocity expressed in RPM (run per minutes);
- **f** is the frequency of the voltage waveforms v_a, v_b and v_c expressed in Hz;
- **pp** is the number of pole pairs mounted on the motor;
- the constant of 60 is needed to convert between the seconds (Hz) and the minutes (RPM).

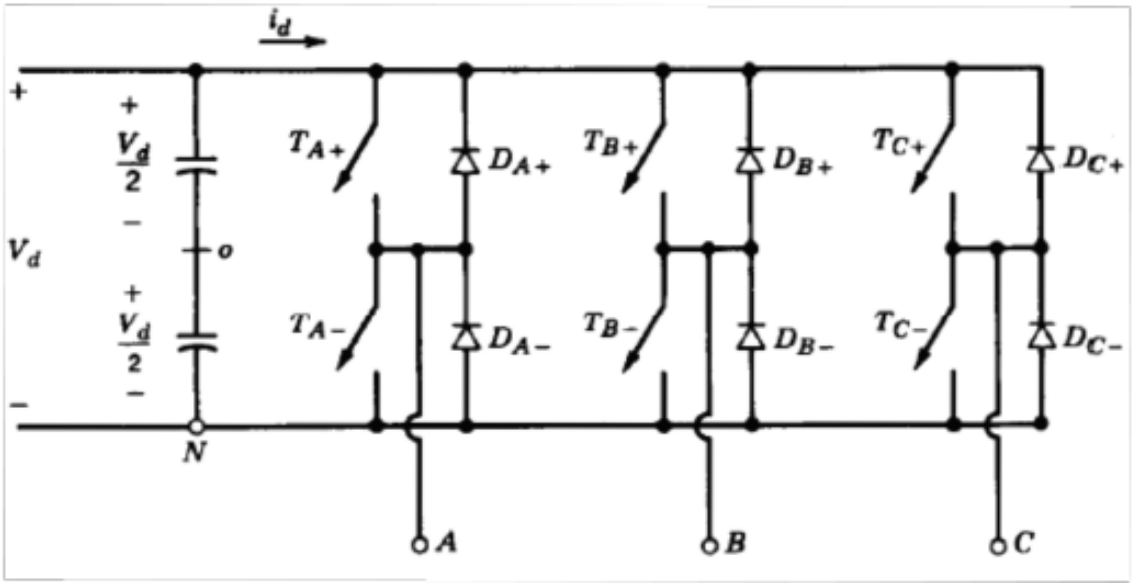

Figure 1.28 A 3-phase inverter generic scheme.[28]

The following Figure 1.29 shows two different schemes for driving an inverter: In Voltage Source Inverters (VSIs), as mentioned, a large capacitor is placed in parallel on the input DC line of the

[28] Mohan, N., Undeland. T., & Robbins, W. (2003), op. cit. p. 254.

inverter to minimize the ripples. Also, sometimes, in case of low impedance load, series reactors can be necessary: one for each branch. As switches, reverse-conducting semiconductor are used.[29]In this configuration, the diodes are connected in parallel to the switches. In Current Source Inverters (CSIs), also named "fed inverters", vice-versa, a large inductor is placed in series and the input current is this way kept constant. In this configuration, the inverter works as a current source and the output current is independent of load, while the voltage depends on it. The commutation circuit is simpler because it contains only the capacitors.

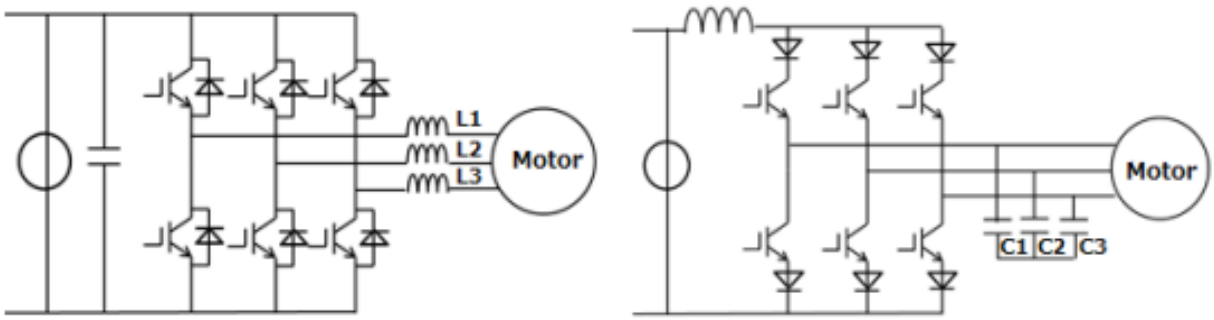

Figure 1.29 a) Voltage source inverter; b) current source configuration.[30]

VSIs are the most adopted solution, especially for home and industrial power applications, this because they can guarantee a smaller voltage drop, but also faster dynamics and controllability.
CSIs are used when the DC input is not constant and changes constantly, they are commonly used in applications as high voltage.

In the next three paragraphs the most used modulation strategies are discussed. Commonly, the absence of a reactor in the VSIs, makes this configuration more compact, but also, as said the modulations strategies are more efficient to be applied. What a modulation does can be shortly described in this practical way:
For 3-phase applications (as will be the one presented on this thesis) the six switches are controlled separately to provide the desired output voltage in terms of magnitude and frequency to obtain the

[30] Toshiba Electronic Devices & Storage Corporation (2018), Retrieved from DC-AC Inverter Circuit: https://toshiba.semicon-storage.com/info/docget.jsp?did=61546&prodName=GT30J341, p. 6.

desired waveform, for each branch the output form is dependent on the DC voltage and the status of the switches, which are related one to the positive and one to the negative, so, they can't be activated simultaneously. All the modulation strategy will be discussed considering their 3-phase configurations.

1.4.1. Six-Step Modulation

The Six-Step Modulation strategy is the simplest one (helpful to understand the basilar mechanism of what happens in a 3-pahse signal modulation) so this method is applied in those applications that don't require high efficiency and precision. Six-step modulated inverters usually causes acoustic noise. The 6 bridges are directly connected to the V_{DC} how it's shown in Figure 1.30 odds are the positive switches and evens are the negative ones. So, respectively to the first branch (A) S1 is opened when the positive voltage is provided and correspondently, the S4 is closed, vice-versa when negative voltage is present: S1 is closed and S4 is activated.

What happens is well described in Figure 1.31, a short delay of a 1 μs must be insert between the commutations of the switches in the same branch, this is necessary to avoid hazardous short-circuit situation.

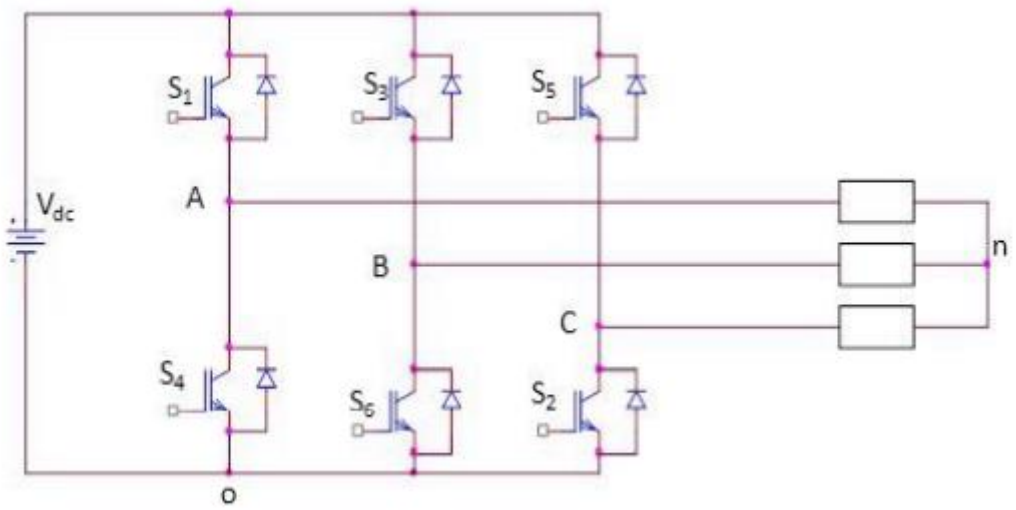

Figure 1.30 Three-phase inverter connected to the load (AC motor), used for six-step modulation[31]

[31] (Rossi, 2018-2019) op. cit. p. 80.

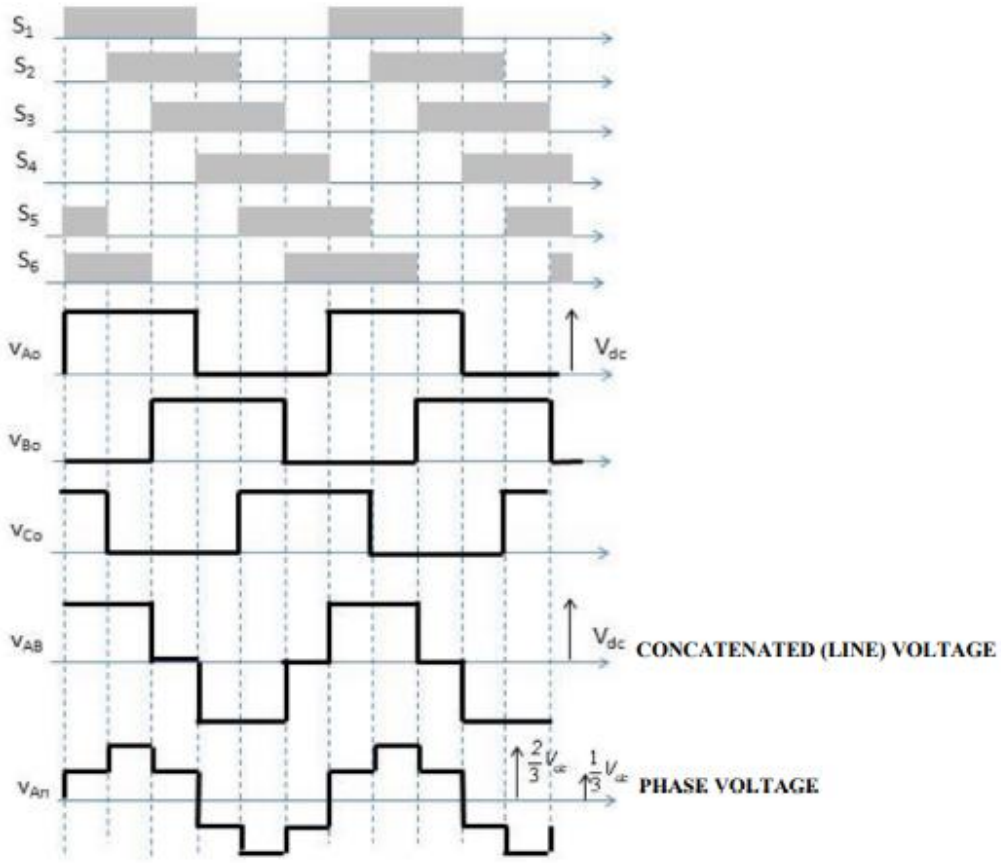

Figure 1.31 Six-step modulation switching strategy and output phase and line voltages.[32]

From Figure 1.31 can be notice that the three command (A, B, C), shifted by 120 degrees one with respect to the others as it happens for every 3-phase system. For realizing a three-phase system simply the concatenated voltages are approximated with alternating waves for driving the electric motor. Fourier series is used for obtaining the first harmonic and the superior harmonics but, the other (higher frequency) can't be detected and this produce a big limitation in terms of precision but also in the noise observed.

Concerning the magnitude modulation: the DC bus voltages directly determinates the magnitude value and this creates a second limitation because according to the theory of scalar control which is based on constant flux approximation (both for induction machines and synchronous one), it is not possible modulate the frequency without modify the magnitude.

[32] (Rossi, 2018-2019) op. cit. p. 81.

1.4.2. Pulse Width Modulation

The mostly commonly adopted modulation strategy for driving electric motors (but also, in general, wherever signals modulation in required) is the Pulse-Width Modulation (PWM). Due to several technological upgrades which determinate a big reduction of costs in terms of hardware, this solution had become the most used one.

The working principle is to have three control sinusoidal signals as inputs (also named “references”) and compare them at each instant with in a “carrier” triangular wave with a frequency at least 10 times greater, this is necessary to avoid bad alias frequencies: when the instantaneous value of the carrier wave is less than that of the single reference wave, the output is positive value (high level), vice-versa when the signal carrier is higher than the reference the output is zero (low level).

Applying this algorithm to the entire signal, on each certain period it is possible to determinate a Duty-Cycle value, of course, for the 3 different signal different duty-cycle are obtained.

For the determinate sampling time each duty-cycle simply “give” the information to each transistor “telling him“ to stay open or close: to make easier this explanation let supposed to have a 0.5 duty-cycle value for a certain time and for a certain signal, this means that for half (0.5) of that time the high level transistor must be open and closed for the other half, vice-versa for the opposite one.

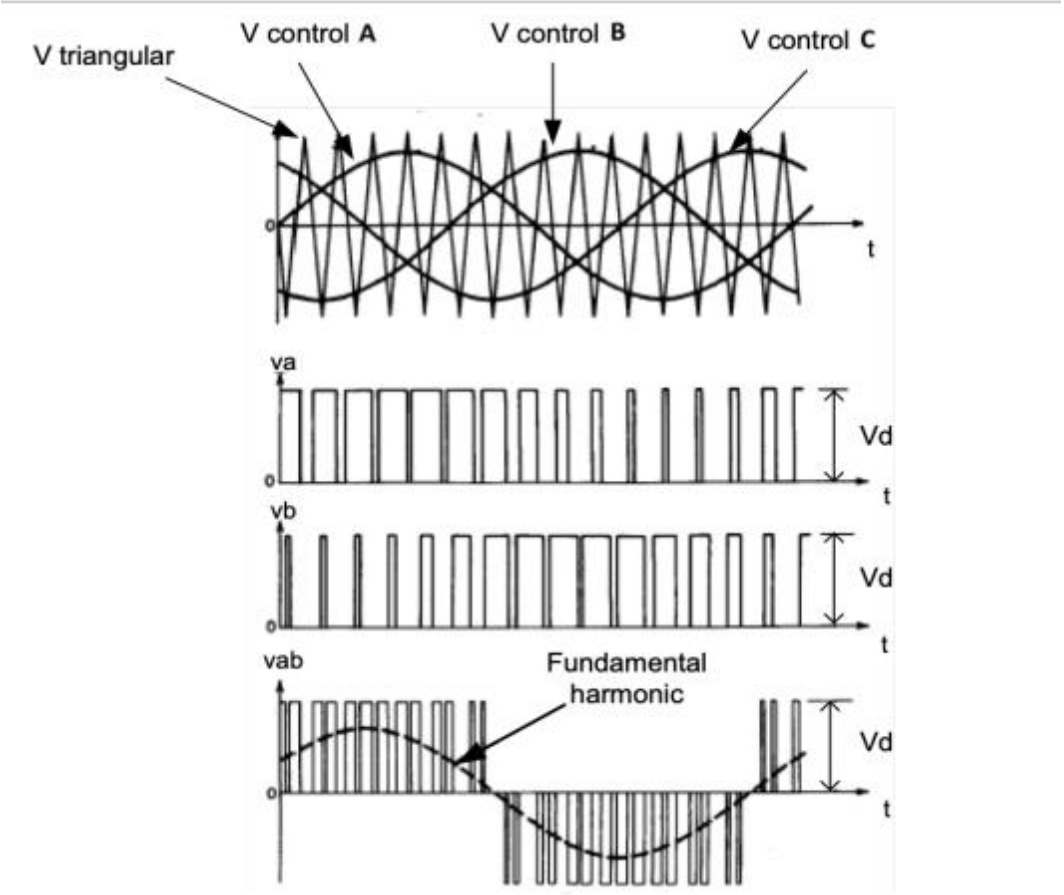

Figure 1.32 Pulse-width modulation command waveforms (modulating sine waves and triangular carrier) and output voltage signals.[33]

Concerning the magnitude, an important parameter for this technique is the modulation index m:

$$m = \frac{V_m}{V_c}$$

Where V_m is the peak value of the modulating signal and V_c is the peak value of the carrier signal. So, by changing m, it is possible to modify the amplitude of the PWM output signal without considering the DC bus voltage value to obtain the desired amplitude value for driving the motor. Indeed, another advantage in PWM strategy is the possibility to also modulate the magnitude of the signal and at the same time kept the V_{DC} constant.

[33] (Mohan, 2003) op. cit. p. 255.

The following two picture taken want to show a typical application of the PWM strategy to a 3-phase signal, in Figure 1.33 a 3-phase signal is reported: the three harmonics are plotted with 3 different colors and they are easily recognized to have 120 degree of phase difference; also the carrier signal with a 10 times higher frequency is reported.

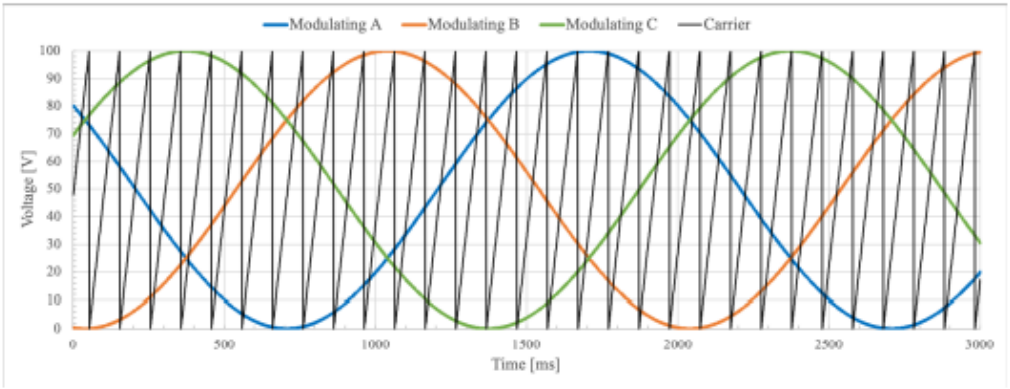

Figure 1.33 PWM three-phase reference waves and carrier sawtooth wave.[34]

In the following Figure 1.34, in red for the A reference the resulting status determined is shown, whereas, in blue, the resulting waveform obtained form the modulation is represented, it's easy to observe that the sinusoidal behavior is maintained, but also the ripples clearly are dependent on the carrier frequency, so, it's possible to expect that if an higher frequency carrier would be available an more smooth output signal could be shown up.

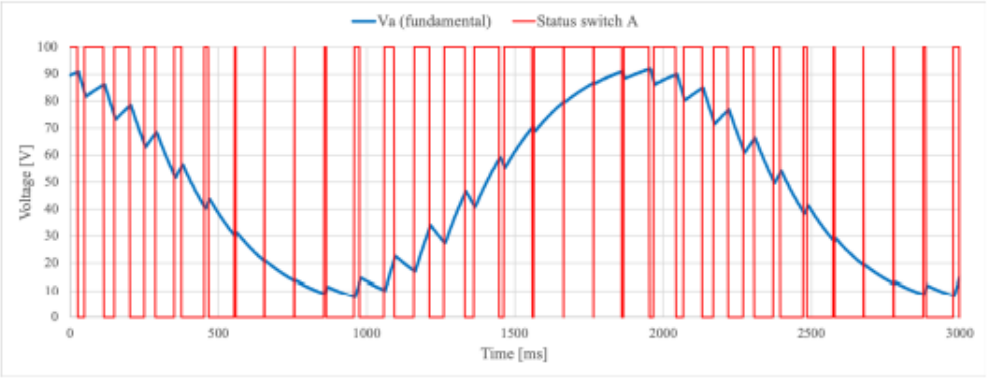

Figure 1.34 Duty cycle of switch A, with the resultant voltage waveform V_a.[35]

[34] (Rossi, 2018-2019) op. cit. p. 85.

[35] (Rossi, 2018-2019) op. cit. p. 85.

1.4.3. Space-Vector Pulse-Width Modulation (SV-PWM)

The model afterward presented is based on the space vector pulse-width modulation (SV-PWM) is an advanced control algorithm, based on the PWM already discussed and commonly applied to induction motors and PMSMs especially when the field-oriented control technique is used. SVPWM is usually used in the FOC to modulate the voltages in output from the inverter, basically this algorithm is applied in the α-β plane, previously obtained through the inverse-Park transform application to the commonly used v_d and v_q waveforms (needed for applying the FOC strategy).

Later, the v_a, v_b and v_c commonly 3-phase quantities are obtained again using both Clarke and Park transformations (αβ to dq and dq to abc). The six transistors where the voltages delivered to the motor; each of the 3 outputs have two transistor: the top closed and the bottom one open, or vice versa. This means that the total number of possible states is eight: simply because $2^3 = 8$.

So, calling each state with a name represented by a vector (form V_0 to V_7) and displacing the 8 vectors in αβ plane forming a sort of hexagonal-star diagram as shown in Figure 1.35, with 60 degree of phase difference between two consecutive vectors. Also, V_0 and V_7 are plotted at the centre of the star they simply represent the full open bottom configuration and the full open top one.

With respect to this configuration for each moment a resultant vector V_{OUT} is computed by the combination of the other states. For example, for a given time T_1 the V_1 is applied, then for a time T_3 the V_3 is applied: to provide the resulting V_{OUT} as a vectorial sum of the two components. So, for each T_{PWM} period a resulting V_{OUT} vector is represented, and their components can be described, associating the given named to the vectors, it's possible to determinate the 6 transistor states.

$$V_{OUT} = \frac{V_1 T_1 + V_3 T_3}{T_{PWM}}$$

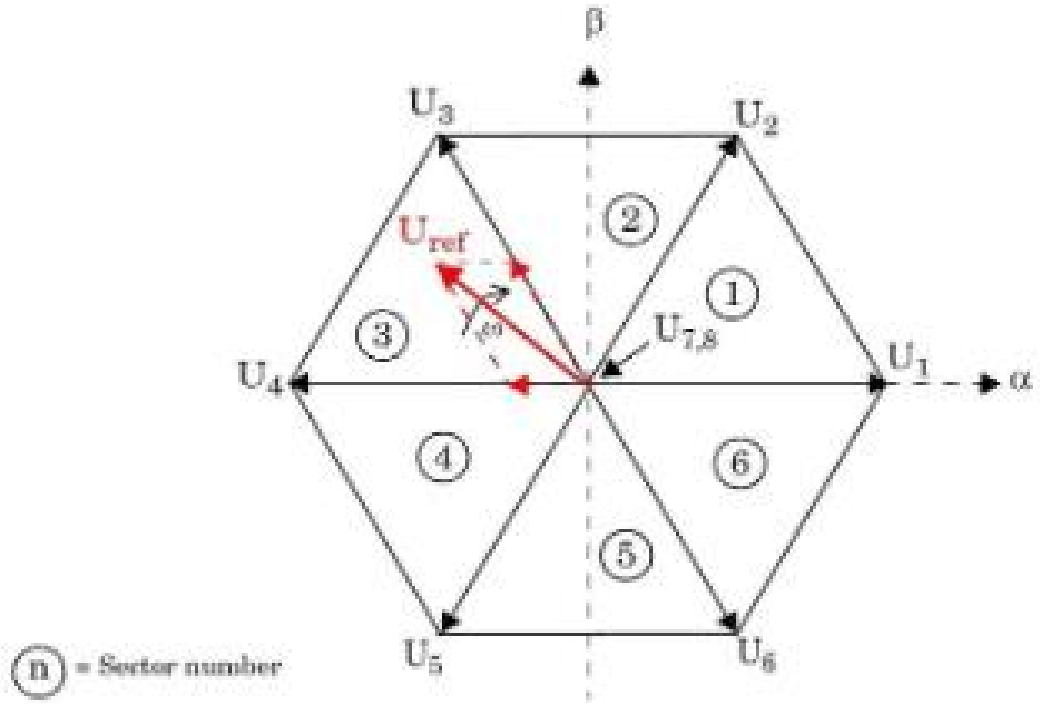

Figure 1.35 Hexagonal-star diagram of the base and null vectors, with the desired output voltage vector[36] (here, the vectors are numbered form 1 to 8, and 7 and 8 are the vectors precedingly called 0 and 7 respectively).

SV-PWM advantages compared to the PWM strategy is huge for the PMSMs applications because the DC bus voltage using efficiency is increased a lot and at the same time the improvement in terms of power factor helps to reduce a lot the hazardous ripples and in general the noise produced.

1.5. Field Oriented Control (FOC) Overview

One of the most used techniques in Electrical drives controllers is the Field Oriented Control Technique (FOC): This is used for AC Induction Motors and Synchronous Motors. As all the variable frequency drives, FOC must be implemented with an Inverter that works with an PWM signals, it

[36] (MathWorks.Inc) La modulazione vettoriale (SVM) per il controllo motori.

also requires transformation of stator currents from the stationary reference frame to the rotor flux reference frame (also known as *d-q* reference frame).

Speed control and torque control are the most used control modes of FOC. The position control mode is less common. Most of the traction applications use the torque control mode in which the motor control system follows a reference torque value. In the speed control mode, the motor controller follows a reference speed value and generates a torque reference for the torque control that forms an inner subsystem. In the position control mode, the speed controller forms the inner subsystem itself. Both inner and external reference are generated comparing the input signal with the feedbacks provided by the sensors which must monitor Real Time currents, speed, position, and Torque.

In Figure 1.36 is shown an overall view of the logic frame of the FOC applied to a PMSM where speed controller is implemented. This scheme will be described in more details in Chapter 3 where all the Simulink building procedure is detailed analyzing one block a time.

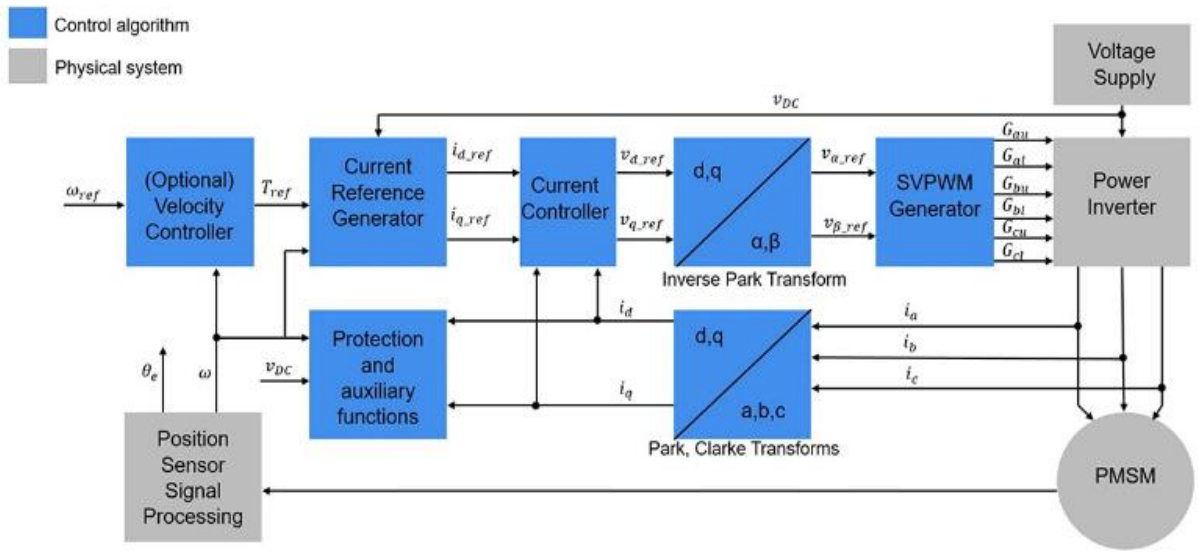

Figure 1.36 FOC architecture for a PMSM.[37]

Field oriented control was introduced during 70s. It is a control mechanism to flexibly drive synchronous and induction motor. It allows decoupling control of torque and speed of AC motors

[37] (MathWorks.Inc) Controllo ad Orientamento di Campo

like separately excited DC motors. Typically, in DC motors armature magnetomotive force (MMF) and rotor flux held orthogonally with respect to each other through mechanical commutation system such as brushes and commutators and this cause limits in DC application due exactly to these mechanical subsystems, typically less affordable than the electric part. In case of AC motors (Synchronous and Induction motor), no mechanical system is needed because spatial angle between rotating stator field and rotor flux changes with the load. This phenomenon causes oscillatory responses which are responsible of using FOC strategy to monitor the rotor field position and orient the stator field accordingly so that angle between both fields can be maintain at 90 degrees. In this way maximum torque condition can be achieved while independently controlling rotor speed (MTPA condition). Stator field is oriented through varying phase and magnitude of three phase ac quantities. Hence it is also referred as 'vector control'.[38]

[38] (Amin F., 2017) op. cit. p. 1.

2. Development Environments: MATLAB/Simulink and Typhoon HIL

2.1. MATLAB/Simulink

This Thesis starts on the building of the control algorithm in Simulink which is a very fast and relatable approach to every single model-based-design problem. Simulink is a MATLAB-based graphical programming environment for modeling, simulating and analyzing multidomain dynamical systems. Its primary interface is a graphical block diagramming tool with a huge set of block libraries crated for an enormous number of applications. For this thesis in particular "Simscape Electrical" and "Typhoon Library" have been used integrated with the elementary MATLAB/Typhoon libraries. It offers tight integration with the rest of the MATLAB environment this is been exploited a lot in the final part of the thesis when simulation results have been plotted together to have an easy comparation between MATLAB/Simulink results and Typhoon HIL Virtual and Real Time results. With a simple script changing just the name of the simulation documents in fact is possible to simulate all the case you want in a totally automatized mode.

Simulink allowed to simulate in the electrical domain both using a continuous a discrete sample mode. For this Thesis it's been used a discrete mode to be more faithful to the aims of the problem that is to simulate Real-Time system with HIL 602+ Hardware. The choice of sample time is also very important because not just the electrical phenomenon requires a different frequency but also because in Real-Time simulation, we will need a performative CPU if we want to simulate in the given time a huge number of mathematical calculations.

Simscape Electrical is a Simscape library which helps to integrated physic-system models with physical connection and integrable with mathematical blocks. Those blocks have been used mostly in the Plant part to simulate the dynamical model of a PMSM, the 6 bridges inverter but is they contains mathematical blocks that simplify the modelling as Clarke and Park transformation blocks (as function directly parametrizable) and SV-PWM signal reference generator the allows to produce within a single block directly from the 3-phase voltages the digital temporized signals for the inverter bridges.

2.1.1. ".TSE" Conversion

Typhoon HIL library can be downloaded from GitHub and quickly installed in MATLAB by running the *file.m*. It works as a "mask" for physical blocks as Electrical Motor, Power-Electronics devices and all the Electrical system available in typhoon, once drugged and dropped in the editor environment you can parametrized your block with all the requirements you want in terms of physical parameter but also the signal you want to receive in output during HIL Hardware simulation. The Motor Block is shown in detail in the following paragraph. So, these "mask" blocks are both implemented with just one click for Typhoon HIL environment and MATLAB/Simulink environment where they are basically substituted with the corresponding Simscape Electrical model blocks (pic…).

Lastly the Block "Convert to .TSE" is used for converting the file to .tse one readable in the schematic editor of Typhoon. The blocks also allowed to directly open the SCADA panel and simulate the model with the Typhoon HIL Hardware, but for some limits in the conversion with this work this feature can't be done (It will be explained in detailed in the results section).

Here is reported the Interface shown by the PMSM block of Typhoon Library, has you can see from the screenshot the block requires you to insert both Mechanical and Electrical parameters, but also the desired output signals, the external SCADA input you want to give (Only in Typhoon of course) and the sensor parameters (Only in Typhoon HIL Real Time mode)

Convert to TSE block is inside Typhoon HIL library section, a special library that can be downloaded from GitHub. These blocks make possible to convert the Simulink Model into a *.tse* file that can be directly used in Typhoon HIL. It's also possible to directly run the code from the Simulink block's interface and open automatically the HIL Scada to run the simulation.
Convert to TSE block also provides a report file .txt that shows which blocks has been converted correctly and the one that can't be converted.

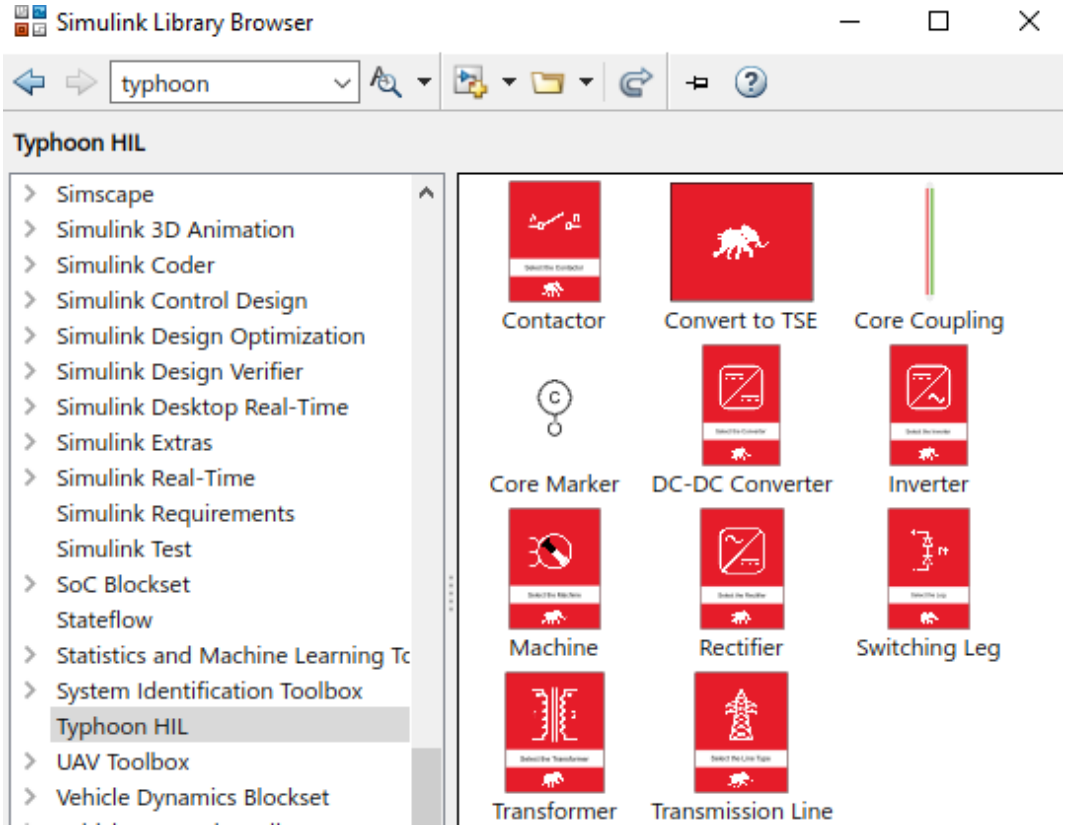

Figure 2.1 This is how the Typhoon library looks like in the Simulink library browser.

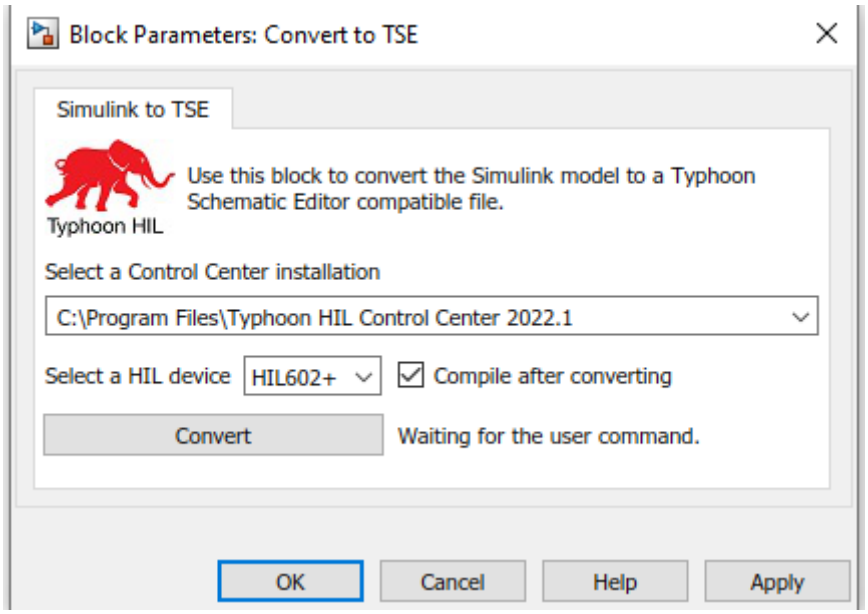

Figure 2.2 The Block “Convert to TSE” in Simulink allow to choose the device.

As result of conversion some differences in the Schematic Editor in Typhoon must be done in order to overcome the conversion limits:

In Typhoon there was no Dynamic Rate Limiter, so it has been implemented by the elementary blocks as in the scheme below.

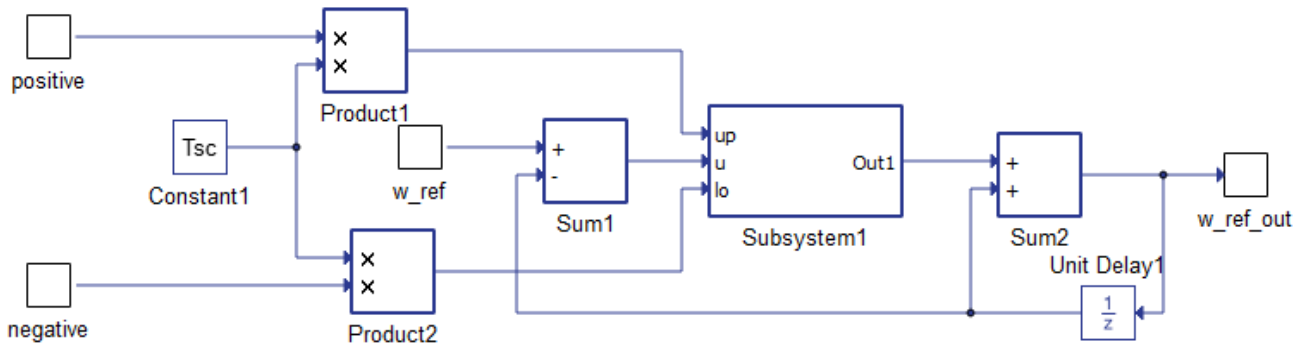

Figure 2.3 Rate Limiter Dynamic implemented in Typhoon, in input it's been passed the reference speed, and the value of positive and negative slope. The output is the reference speed adapted instantaneously with the slope limitation imposed from the BMS.

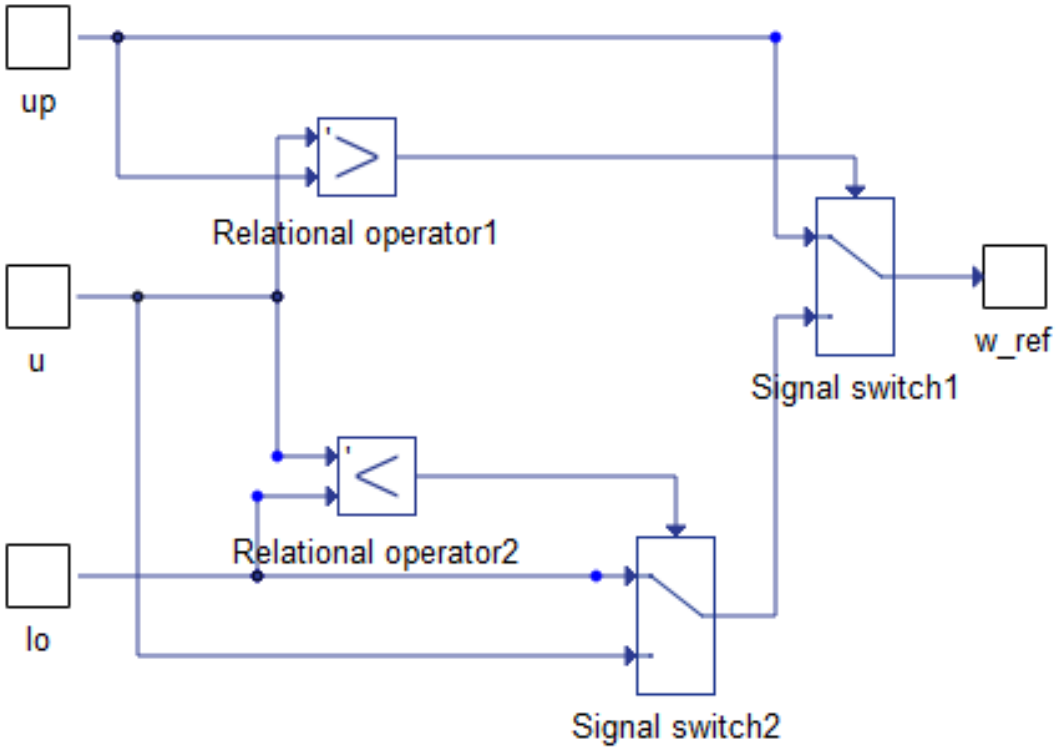

Figure 2.4 This represents the subsystem1: here the correct relative slope is chosen between the 2 positive and negative value.

The main difference is in the SV-PWM strategy, to be clearer as possible below are shown the two schemes in Simulink and Typhoon. The main difference is in the block's "PWM Generator" output

which are two different kinds of signals: in Simulink these are 6-digital signal that activate the 6 bridges inside the inverter (3 for positive and 3 for negative value). Otherwise in Typhoon the block is replaced with a "SV-PWM reference generator that provides a 3-phase duty-cycle signal which will be seen from the inviter as

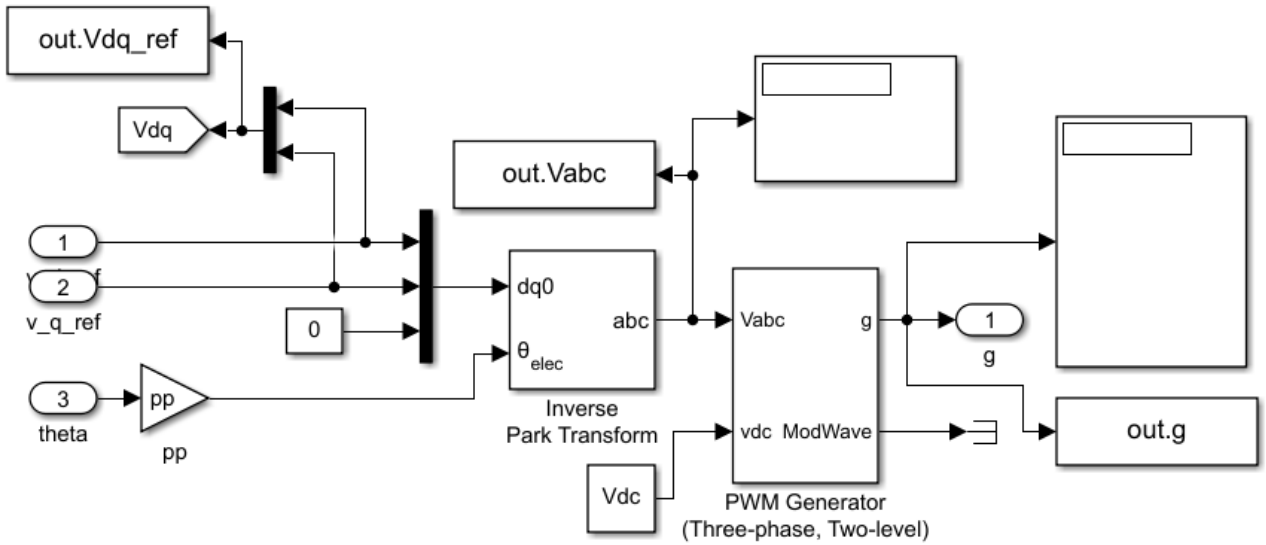

Figure 2.5 This is the Simulink scheme: the main difference between the 2 schemes is that the inverter switching frequency is defined inside the block "PWM-Generator".

The block provides a triad of value for each step of the simulation where alternatively the positive or negative bridge of each phase is activated (positive) or disactivated (negative).

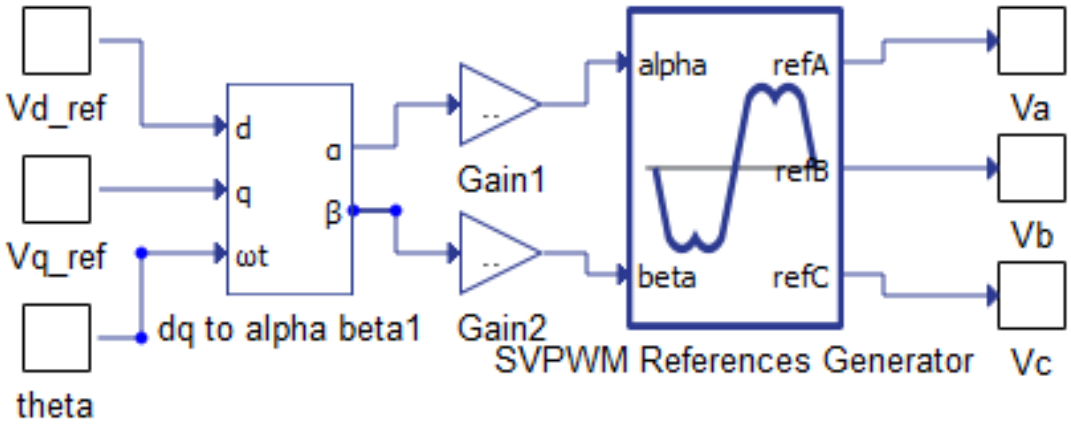

Figure 2.6 Typhoon scheme: Since the output must be a normalized signal between 0 and 1 the Voltages must be normalized before the SV-PWM generator by dividing for 230.9V (V_sat_phase).

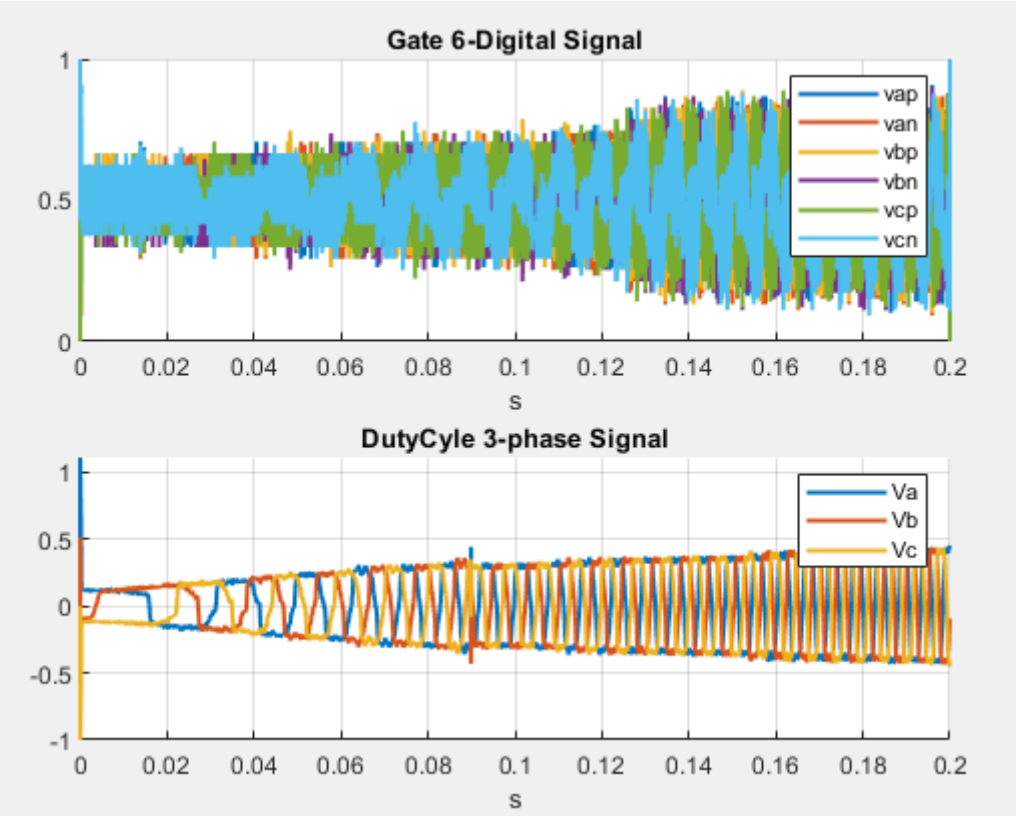

Figure 2.7 This is a representation plotted in MATLAB of the 2 types of signal. a) The Simulink 6-Digital signal. b) Typhoon Duty-Cycle signal of 3-phase voltages.

There are also many little issues to overcome in the conversion phase that practically makes impossible to directly launch the computation, but they can be easily overcome paying particular attention during the building phase, a short list is reported below:

- Go-To/from;
- Bundles or Bus Selectors;
- Subsystems;
- Input/output of each Subsystem;
- No square function (x*x instead of x^2);
- Scopes/Probes (They are not necessary);
- Switches;
- Clarke and Park transformations;
- Others not so important;

2.2. Typhoon HIL: Hardware in the loop

2.2.1. Hardware

Typhoon HIL 602+ is the hardware used for this work, but there are also many others Hardware developed by Typhoon. Hardware in the Loop technique is used in the development and testing of complex real-time system. The complexity of the plant is replaced with a mathematical representation. The principal advantage is the possibility to reduce cost, duration, safety, and feasibility compared to a real prototyping test.

2.2.2. Software and Simulation Interface

Typhoon HIL Control Center (THCC) is the software provided by Typhoon that can operate together with the Hardware. Also, another advantage is the possibility of a virtual mode simulation. THCC present 3 main features:

Schematic Editor: is the environment in which you can build your model through a huge number of components and power electronics blocks. It presents a huge library developed for many different applications related to electric, electronics, electromechanical and power electronics systems. Typhoon also provides a big documentation where so many models are already uploaded by the developers and constantly updated every few months.

The HIL Scada: is simple graphical interface for manage the simulations in which you can easily build your own specific interface with many tools already built but, most important quickly imported with the use of python code generator functions (API);

Typhoon Test IDE: which is the framework in which you can write and test your python code for the complete simulation of just for develop your own API functions.

2.2.3. S-Function Building

First step is to set a single subsystem for the entire controller part in the Typhoon model, in this subsystem all the controller computation is elaborated from the CPU.

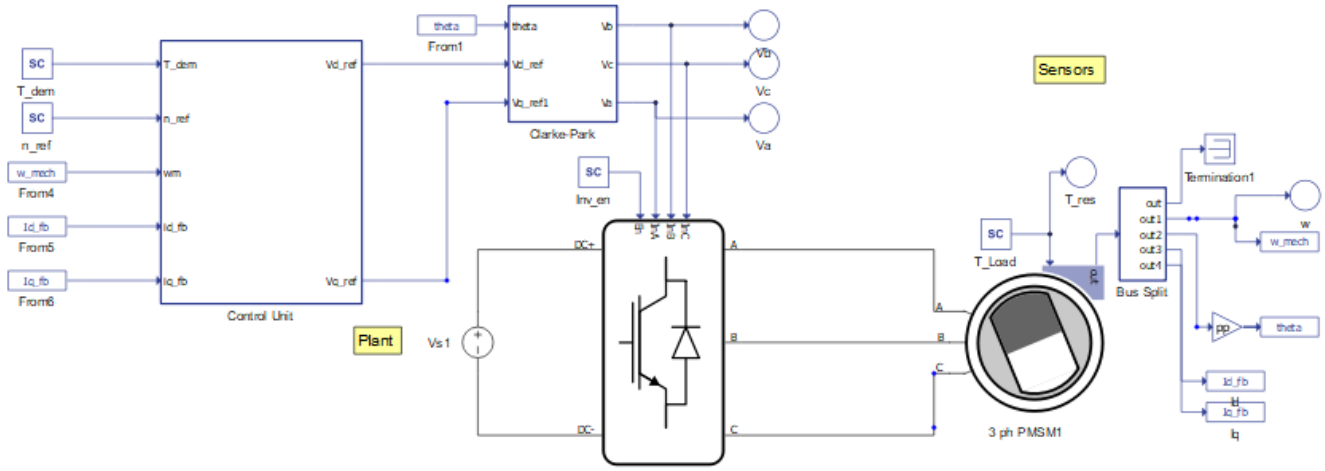

Figure 2.8 The Complete Model with single “Control Unit” block.

As it’s clear from Figure 2.8 the inputs in the Control Unit block are five: 2 directly set “real-time” by the user and 3 form the feedback sensors. The 2 outputs are the voltages V_{d_ref} and V_{q_ref} that will be exploited to generate the SV-PWM signal to the Inverter.

As mentioned in Chapter 2, Typhoon provides the possibility to generate the C-Code form a subsection of the Model. The Generation consist of a generation of a cartel where three files are automatically generated:

- The .C file where the computation algorithm is inserted;
- The .H file is the Header file through which the Data structure are defined and settled;
- The Types.H file is where the types are defined.

Of course, the 3 files are called in cascade.

Once the Code is correctly generated it’s necessary to set an S-Function in MATLAB that can read the inputs pass them to the C-Code and once elaborated pass again the outputs to the rest of the Simulink model (practically the Plant). This S-Function is compiled only once and then its name is passed to the Simulink block that exactly replace the Control Unit block in Typhoon presented in Figure 2.8. The Complete Code-Generation is discussed in the Appendix.

2.3. LabVIEW

LabVIEW is National Instruments (NI) a software that uses a a visual programming language for development, design and testing of systems concerning a huge range of applications. The principle

of working of LabVIEW is not much different from Simulink's one: It integrates subroutines, named Vis (Virtual Instruments) composed of three elements: a front panel, a block diagram, and a connector pane. The first two are directly linked each-other and they can be seen as the two faces of the same medal: in the front panel stands the controls and so it is used for controlling the running-simulation; the block-diagram otherwise is where the system can be visualized and manipulated, so it is used especially during the build phase. It's important, for the ones not familiar with LabVIEW that every block (or sub-VI) is always visible in both areas in two completed different ways to interact. The connector pane simply is the way with LabVIEW can access to every sub-VIs present and insert them inside other Instruments. In this object is very important to well indicates the inputs and distinguish them from the outputs to use the sub-VI as "connection function" between to different blocks.

LabVIEW uses a graphical language called "G", which is a very intuitive approach for high-level programming languages where simple-representations of laboratory items can be just drag-dropped and linked each-other inside the front panel. As mentioned before the items once placed inside the front panel also are represented inside the block-diagram where the programmer can easily drag-drop and link the different native blocks (or sub-VIs) with mathematical/logical and other sub-elements to express the program the is wanted to be build. LabVIEW also allows interfacing with many external or third-part different devices, instruments for measurement, vision systems, etc. For this project this approach has been used to complete the closed-loop Hardware in the Loop system. Of course, the correct version of each external device's driver must be properly installed on the host PC and the right Operating System must be used to be able to let the compiler understand both native languages. NI provides a very detailed on-line documentation to better understand how to approach the problem also for the nonprogramming-familiar users. A compiler provides the latter functionality: the executable machine code is generated in this way, translating the graphical code according to hardware native functions.

At this time, to better going into the very specific example: Teoresi Group SpA provided a-third part embedded electronic circuit named "ECU SPARK" from ALMA AUTOMOTIVE which has got a Linux-Real-Time-Operating-System. (Linux RT OS) which can reach very high performances in terms of speed of calculation, this will be necessary in the very case because other common Widows-OS can't be fast enough to guarantee the Real-Time computation of the controller. The SPARK device is piloted directly through a friendly break-out board that can be easily connected to the host PC (via ethernet protocol) and easily wired to the Typhoon HIL 602+ integrated breakout board.

To let easier understand all the system built in now reported a short list of the devices, software and their specific versions or configurations:

- LabVIEW 2020 SP1;
- LabVIEW FPGA Module 2020;
- LabVIEW Real-Time Module 2018;
- NI CompactRIO, NI-RIO, NI LabVIEW FPGA Xilinx Vivado 2019.2, are the drivers needed.
- Windows 10 OS on the host PC;

Two more necessary tool are needed to correctly build the system:

- A Secure-File-Transfer-Protocol (SFTP); in this case WinSCP has been used.
- A C-Code manipulation environment; in this case Visual Studio Code has been used; to build the .iso library to export C-Code inside the Linux OS also the C-Make extension and the Ninja Cross-Compiler are needed.

2.3.1.“.SO” Library Building and placing inside Linux Filesystem

First thing to build is the .so library that will be re-called inside the block-diagram node “call a library function node”. For this process is really suggested to use the NI tutorial guide[39]. Basically, what is needed is a structure readable from the Linux RT OS exactly in the same way of a common windows .dll library can be read from a Windows OS: at this scope the Ninja cross-compiler is needed as explained in the NI tutorial guide. Inside this structure in the project cartel created in Visual Studio Code all the C Code generated from typhoon must be placed. Also, as better explained also during the S-Function building paragraph, the wrapper files must be used to overcome the problem of the complicated structure of data automatically generated from typhoon.

As the .SO File is created it must be placed inside the correct path used from the Linux OS when it runs the RT simulation trough LabVIEW interface. To do this, in this case, the WinSCP software has been used to have access to the Linux Filesystem and very easily drag and drop the file inside the correct path: Usr\Local\Lib which is the path followed by the OS as the simulation is run.

[39] https://nilrt-docs.ni.com/cross_compile/config_dev_system.html.

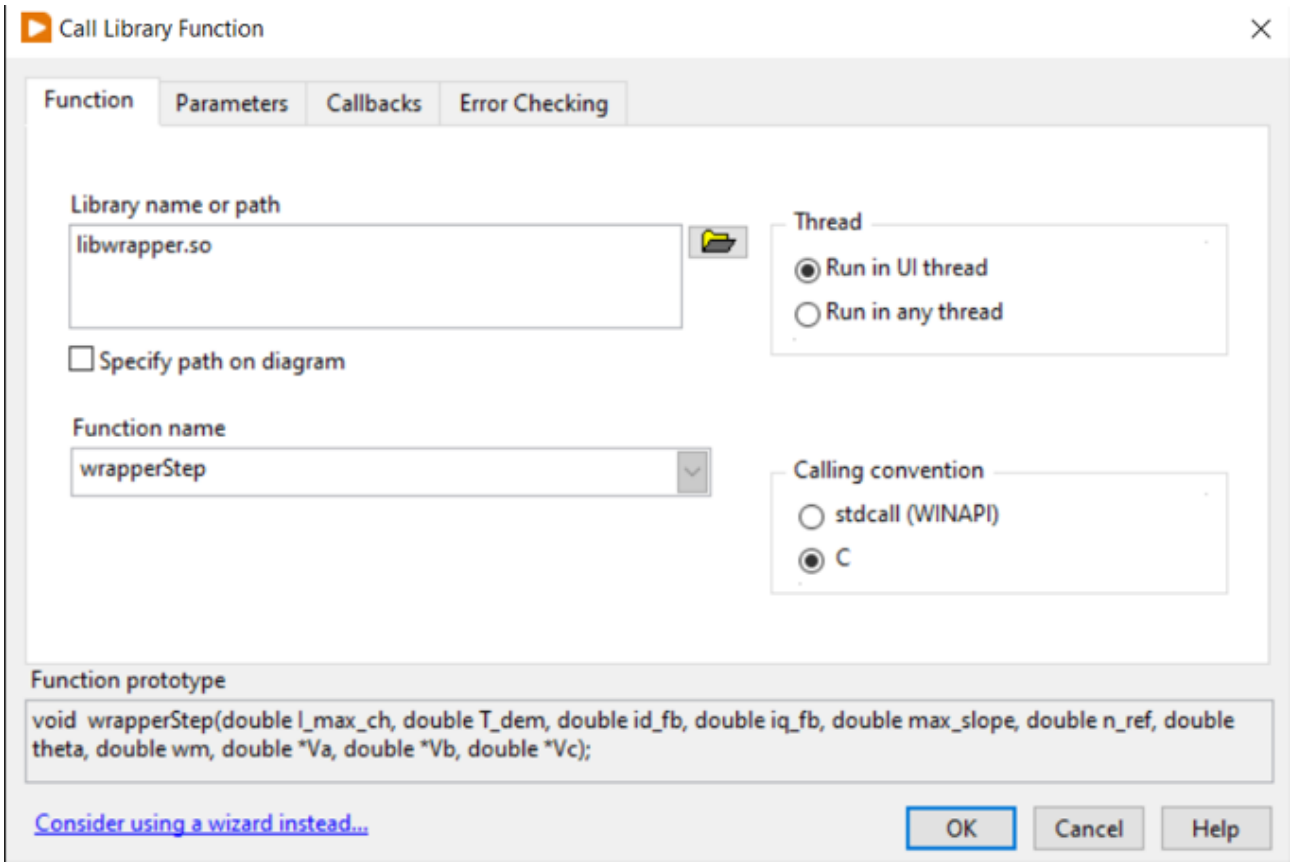

Figure 2.9 Call a Library Function Node configuration. In the bottom of the screen-shot is clearly shown the wrapped function with all the parameter passed as inputs and outputs.

2.3.2. FPGA and Real Time Module using

Both Real-Time Module and FPGA Module adds-on are used to run the controller code inside the Linux RT OS and meanwhile read and write by/from the analog signal that interacts with the Plant part (simulated by Typhoon HIL 602+ device and operated form the SCADA Panel). In this case the FPGA module can't be properly used for his native purpose which should be the computation of the plant (it can reach 40 MHz) but in this specific case it necessary to run the controller code with the LabVIEW RT Module which can reach a slower speed (1 MHz). In any case the FPGA Module must be used the same because it necessary to write and read the output/input analog signals from the breakout board.

As a first step is necessary to explain the process to follow and a general overview of the architecture used. In Figure 2.10 is reported the three of the LabVIEW project:

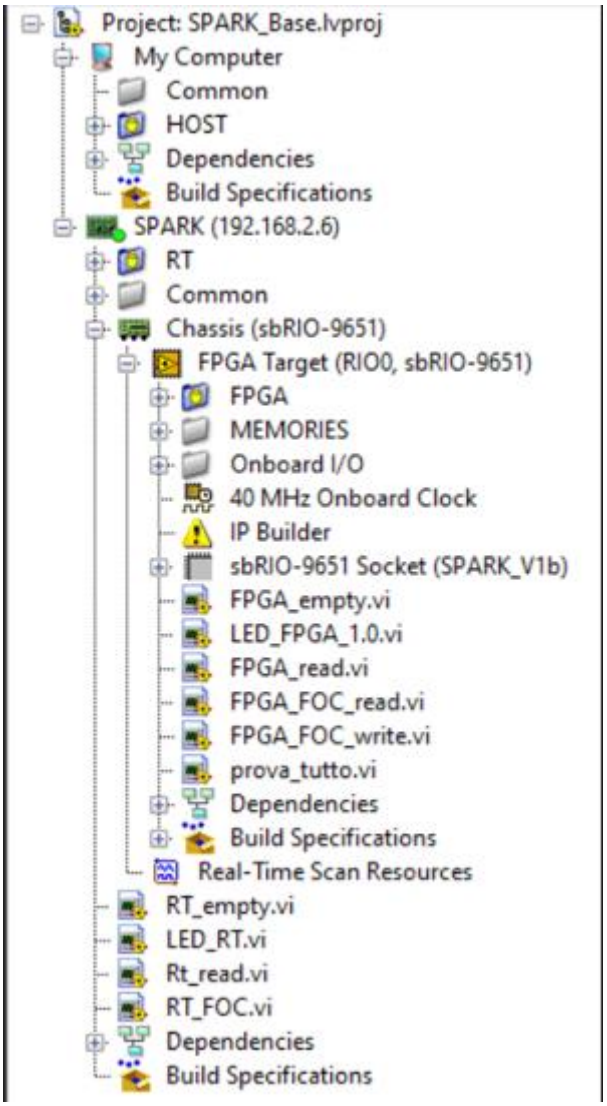

Figure 2.10 The project tree that show the general architecture of the system.

It's clearly visible that the FPGA Target has a "red light" turned-on which indicates that the Target relates to the host PC (via ethernet) also is necessary to use the correct IP configuration in order to make possible the two devices to communicate. Once the Single-Board Reconfigurable I/O (sbRIO) Target is connected inside the Target it's possible to see a file name "socket CLIP" which stands for component-level IP which basically is a .xml file which is used to make the OS to read and understand the breakout board's pinout. So, the CLIP file assigns a specific name to every single pin. It's possible to create your own CLIP file using a wizard provided by LabVIEW FPGA Module, in this case the CLIP used is provided by the company that also provided the hardware (ALMA Automotive itself).
The Second step to understand is to configure the FPGA VIs and the corresponding usable RT Module VIs: As mentioned, the first one is a VI that directly run inside the FPGA on the board, the second one is Real-Time executed form the RT OS.

Blinking Red Light Example

Before going on presenting the Real-Time application it can be helpful to introduce a practical example to better understand the behavior of the Real-Time architecture. The following example has been built to demonstrate how the FPGA Module and the RT Module operates together. The Red LED on the ALMA SPARK board is one of the pins presents on the pinout file. In this case the pin doesn't need the breakout board to be piloted because the Red LED is directly mounted on the board as showed in **Error! Reference source not found.**. Being an only readable pin, it can't be used as an i nput.

Figure 2.11 ECU SPARK by Alma Automotive. The Red LED is placed where the red arrow indicates, and it can be turned on and off by the user or the running algorithm

In Figure 2.11 it is shown the FPGA VI that enables the Red LED and, inside the while loop let the user settle the LED status (ON/OFF).

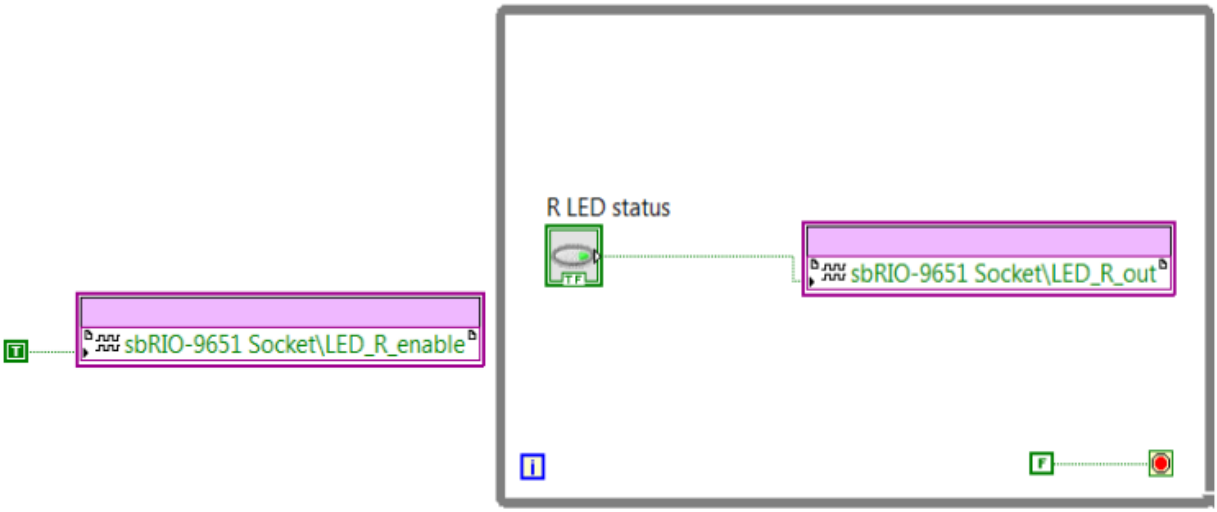

Figure 2.12 Block Diagram of the FPGA VI example.

To let the FPGA VI to be usable first it must be compiled and deployed in the SPARK ECU. Once deployed the build file can be directly lunched without any compilation. Also, The FPGA VI can be used as a sub-VI, exactly as said before, and most important can be re-called by a RT VI. In

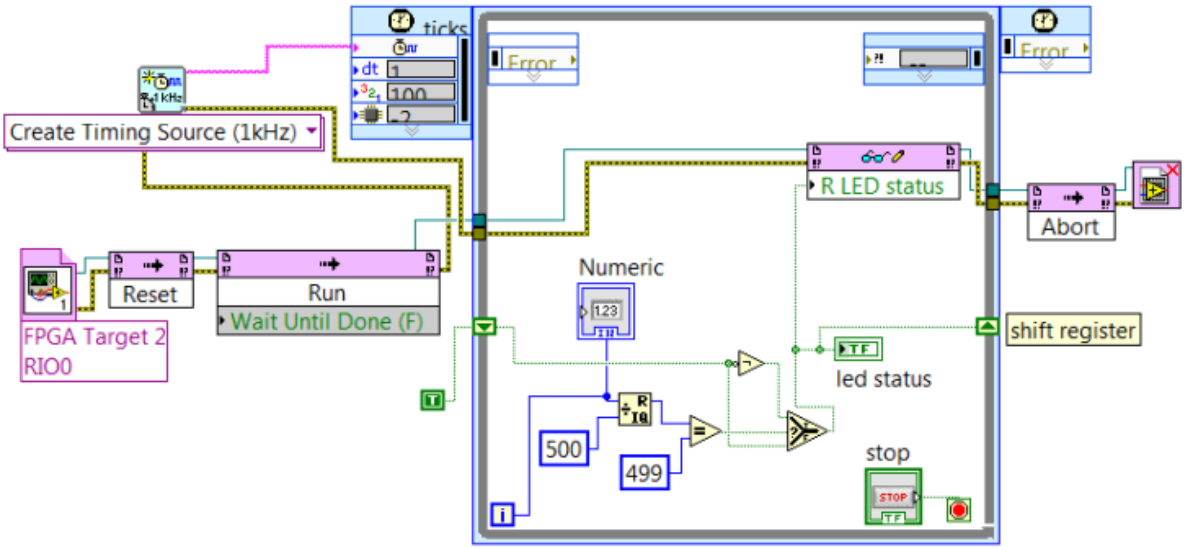

Figure 2.13 is illustrated the RT VI for the example.

In this example the first block from left is settled to open the FPGA VI deployed on the Target, than the VI is “run” and inside the loop the simple mathematical algorithm let the LED status be true only when the condition expressed is true, so changing the timing source on the left is possible to accelerate or the decelerate the blinking frequency (In this example the LED blinks once every 500 ms).

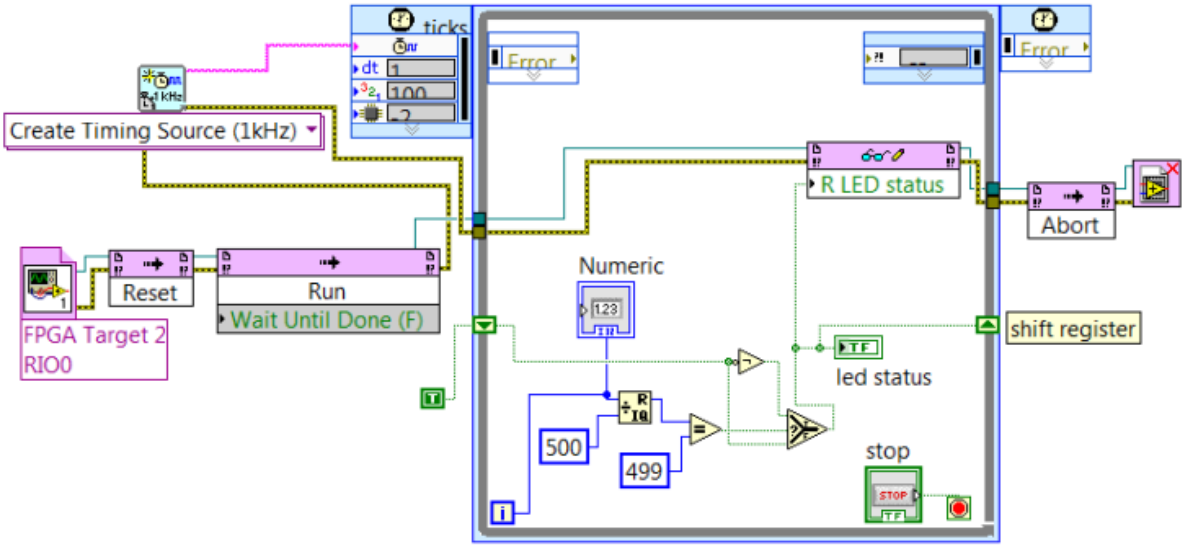

Figure 2.13 Block Diagram of the RT VI where the FPGA VI is re-called and used in loop to have an algorithm the control the blinking of the Red LED.

The result of the mathematical computation is “write” on the output LED (Red LED) and also visible in the front panel. In the end the method open is aborted and the reference to the FPGA VI is closed.

3. FOC algorithm Design

3.1. First Presentation of the Model

In this chapter, the design of the field-oriented control algorithm is discussed. Moreover, in the final part the battery management system (BMS) is implemented as safety criterion that can works as a bottleneck of all system i.e. this BMS will be the most external loop in the algorithm. The direct current limits, due to battery overcharges or malfunctions are chosen by the user. The control method is designed in an ideal condition where no sensor architecture is designed, this because the model of sensors will be replaced by the HIL 602+ directly when we simulate the algorithm in Real Time on the Hardware. The rotor electrical speed and electrical angular position are passed to the controller by using local variables in Simulink environment, so the encoder and Hall effect sensor are neglected in this situation. Initially, the schematics of the controller and plant are presented, showing their general structures. Then, the blocks are analyzed individually, describing the main features and the adopted solutions for their realization. The different PI regulators, the maximum torque per ampere area and the field-weakening region are described, highlighting the optimization tools for increasing the dynamic performance and the maximum deliverable torque. The MTPA is a combination with 2 different input strategies: the reference torque can be directly passed for the current's calculation, or the speed error can be controlled by a PI and then is converted as the reference torque. The 2 method can be chosen by the user depending on the system condition: the drive style for example but as example in the following results will be simulated with a speed threshold of 150 rad/s that will determinate the switching condition between "Direct Torque Reference Strategy" to "Error Speed Compensated Strategy"; this because to control the speed, commonly called "cruise control" is usually done in regime speed situation at high speed as can be an highway. In figure (pic), an equivalent complete block diagram is represented. It is useful for visualizing the inputs and the outputs for each element, the exchanged values between the controller and the interior permanent magnet synchronous motor and the status variables.

The interior permanent magnets synchronous motor is able to guarantee a larger power density and a higher maximum torque with respect to a SPMSM in the same operating condition: this advantage is related to the presence of saliency in the airgap. In fact, the equivalent direct axis and quadrature axis stator inductance – respectively L_d and L_q – are different (in detail, $L_q > L_d$ because the d-axis is aligned along the magnetic flux direction), so the additional reluctance torque can be exploited. In

the design process, first of all the model of the plant is developed, by using the generic PMSM mathematical model (analyzed in chapter II). Using the dynamic equations, the delivered electromagnetic torque can be computed, also considering resistance and damping effects. The model of the motor is inserted in a while loop, and it uses a different discrete time interval for the integrators with respect to that of the controller. Concerning the field-oriented control, initially the Clarke-Park transformation blocks are explained, they basically convert from the three-phase stationary reference frame to the two-phase rotating axes. Then, the proportional-integral regulators are analyzed and designed, removing all the possible problems related to saturation and windup. Also, some saturation blocks are used, and it will be explained each choice in term of physic meaning and how the saturation value It's been chose. Once established the core of the FOC, the reference values of the currents are obtained through the MTPA block. The Speed reference signal is passed as a ramp signal this because the stability and the ripples of the dynamic response of the system is strictly depended on the slope of the ramp and It will show in the results. For the field-weakening region and the regulation of direct current a feedforward strategy is used. Lastly according with safety criteria, the actual I_{DC} is computated and an % error as output of the BMS block is generated in order to reduce maximum slope available with the I_{DC} chosen. In the following table a list of parameters used is report: motor parameters, simulation parameters and controller settings are reported. A star-connection (Figure 3.1) has been considered for the three-phase stator, so the relations between phase and line quantities are:

$$V_{line} = \sqrt{3} V_{phase}$$

$$I_{line} = I_{phase}$$

$$P_e = 3 V_{phase} I_{phase} \cos(\varphi)$$

where cos (φ) is called power factor and it represents the angle between current and voltage phasors, while P_e: is the total active electric power of the system – obtained through the sum of the three individual contributions (one for each phase).

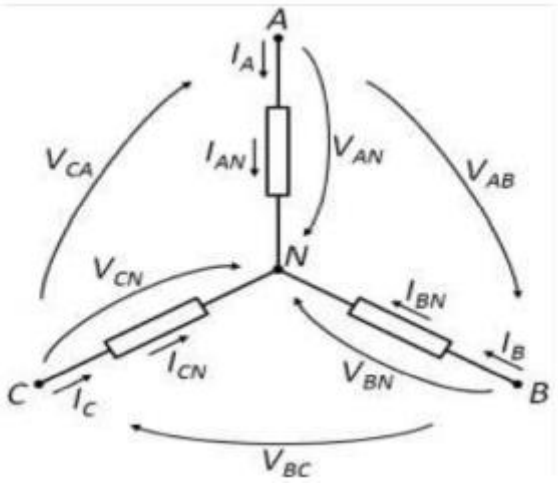

Figure 3.1 Star-connected three-phase stator windings.[40]

The following Table recaps all the physical parameters used in the design of the model, some of the parameters presents more than one value due to the fact that there are been exploited in the simulations as changing parameters in order to understand how the model responds to different input. This will be clearer in Chapter 4 where simulation results are discussed.

Name	Symbol	Value	Unit of Measure
d-axis inductance	Ld	215	μH
q-axis inductance	Lq	86	μH
Pole pairs	pp	5	.
PM flux	λ_PM	0,044	V s
Moment of inertia	J	650,2	Kg cm^2
Dumping factor	B	0,076; 0,114	N m s
Stator resistance	Rs	8,5	mΩ
DC voltage	V_n	400	V
Saturation phase voltage	V_sat_phase	230,9	V
Saturation phase current	I_sat_phase	485	A

[40] Ferreira, F., Ge, B., Quispe, E., & De Almeida, A. (2014), Star- and Delta-Connected Windings Tolerance to Voltage Unbalance in Induction Motors, Presented at International Conference on Electrical Machines (ICEM), Berlin, Germany, p. 2

Nominal Torque	T_nom	145	N m
Maximum Torque	T_max	237	N m
Maximum Slope	slope	600,900	rad/s^2
Nominal Speed	n_n	6000,9000	rpm
Power factor	cos(φ)	0,94	.
Max Charge Current	I_max_ch	300,500	A
Max Discharge Current	I_max_dis	-300,-500	A
Resistence Torque	T_res	1	N m

Table 1 All the parameter used for the model design.

3.2. Plant

3.2.1. PMSM Model

For realizing the plant simscape electrical library has been used, here is shown how Simulink develops the mathematical system through elementary blocks: the starting point is the dynamic model of the IPMSM, analyzed in the Chapter 1.2. The command variables of the motor are the three-phase stator voltages provided by the 3-phase inverter, which are transformed in the corresponding d- and q-axis values and then used for computing the direct and quadrature components of stator current. In this operation, the values of stator resistance R_s, direct and quadrature inductances L_d and L_q, permanent magnet flux linkage λ_{PM} and rotor electrical speed ω_e are used. Most relevant thing of course is the Torque computation for wich we remind the equation:

$$T_{em} = \frac{3}{2} pp\left[\lambda_{PM} + \left(L_d - L_q\right) \cdot i_d\right] i_q = T_{em,syn} + T_{em,rel}$$

There is also a block called “Hall effect Sensor” that is internally used to compute the positioning and the speed exploiting the variation of the angular position. As we said, these sensors are considered neglected because no mathematical dynamic model has been implemented; In typhoon the hardware is able to also simulate the sensors dynamic.

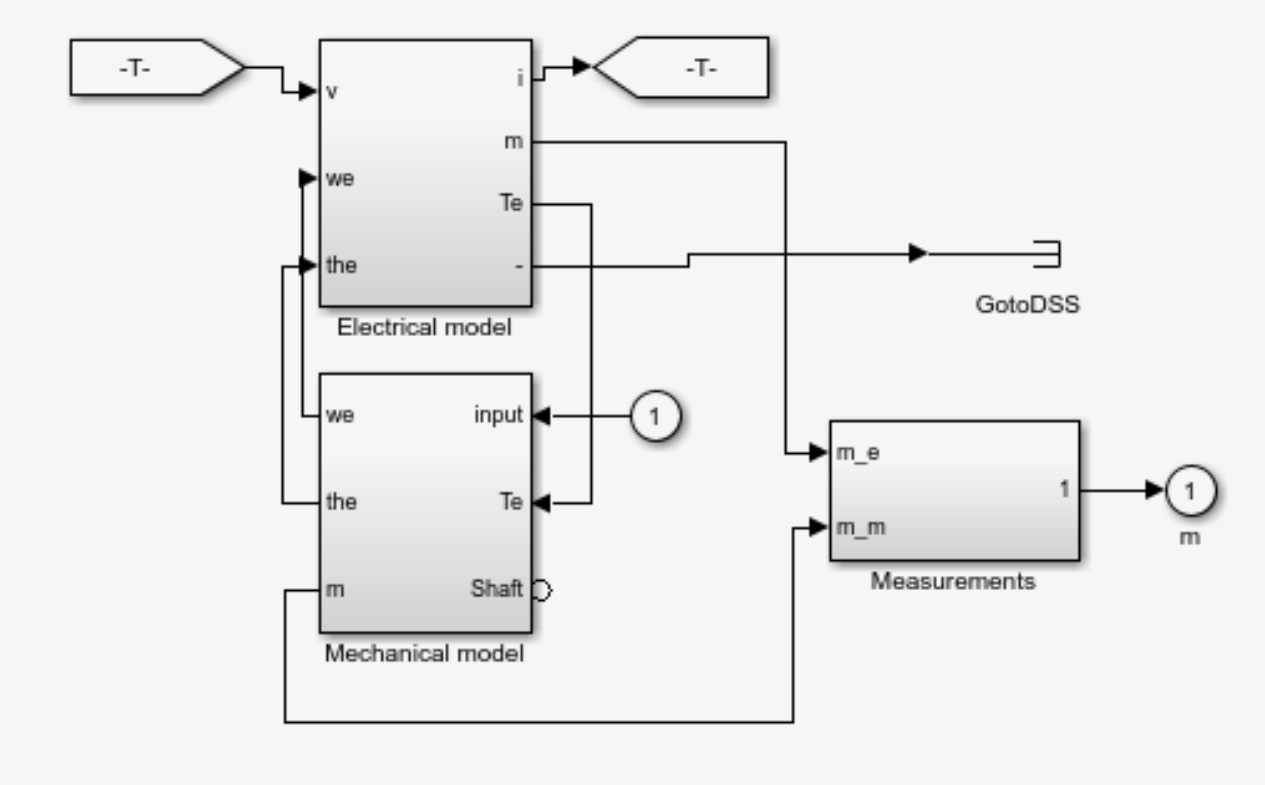

Figure 3.2 Overall scheme of the Block PMSM in Simulink

Once obtained the actual i_d and i_q, they are transformed into i_a, i_b and i_c through the inverse Clarke-Park transformations and passed to the controller. In Figure 3.3 the Electrical Model block is fully represented in detail.

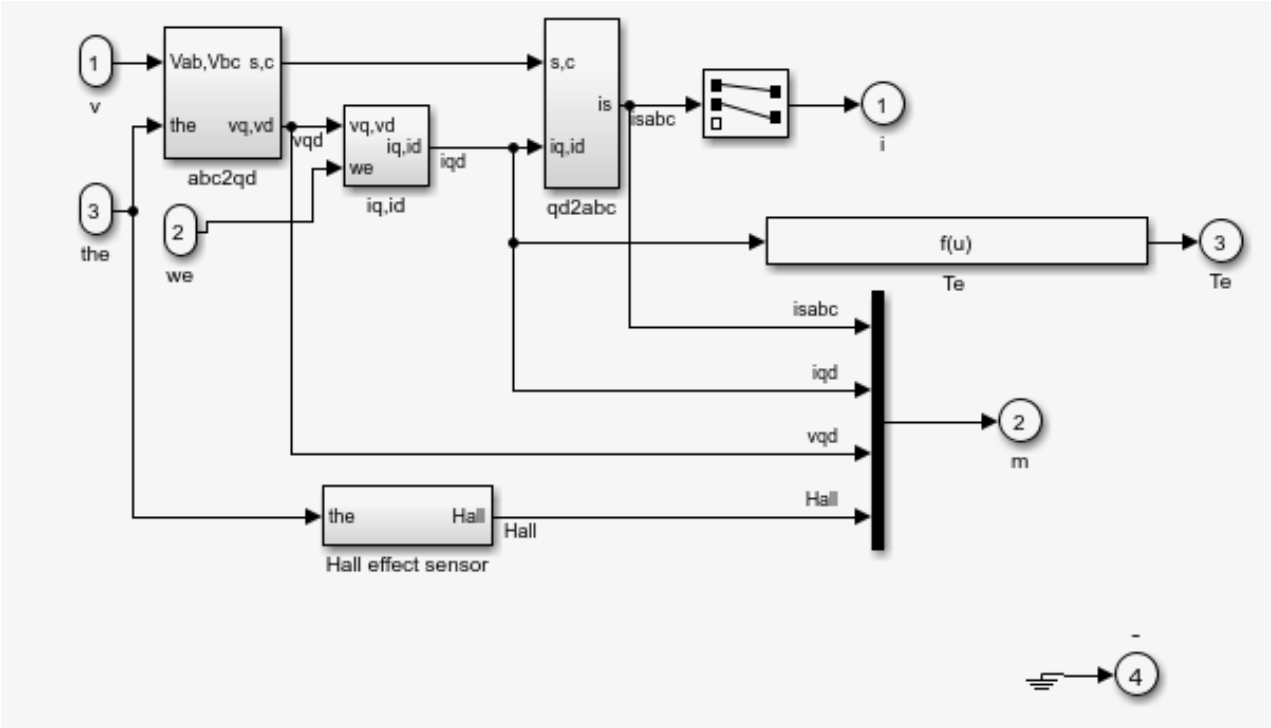

Figure 3.3 Electrical Model of PMSM

The other main block is the Mechanical Model: in this block basically, the speed is evaluated. Here is reminded the mechanical equilibrium of the rotor torque. Subtracting from T_{em} the constant resistance torque and the variable term depending on the actual mechanical speed and on the damping factor γ the mechanical angular acceleration is calculated. In the successive step, for simplicity, the integrator output is the electrical speed, obtained by multiplying the inverse dynamic equation for the pole pairs:

$$\begin{cases} \omega_e = \dfrac{pp}{J}\int \left(T_{em} - T_{r,const} - \gamma\omega_m\right) dt \\ \omega_m = \dfrac{\omega_e}{pp} \end{cases}$$

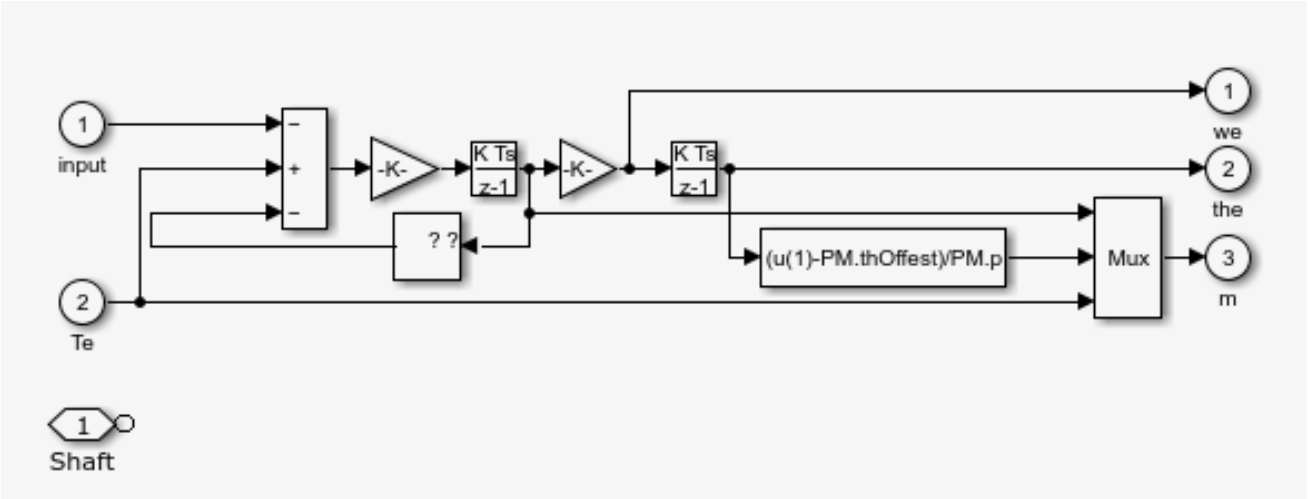

Figure 3.4 Mechanical Model of PMSM

3.2.2. 3-Phase Inverter SV-PWM Modulated

The following picture shows the detailed working principle of the 3-phase inverter implemented in simscape electrical library. It's particularly clear that the inverter has got 6 IGBT diodes that are enabled simultaneously by the SV-PWM signal that determinates when the positive port is active and the negative one is closed and vice-versa. Each IGBT diode is internally modelled in order to provide also the right reference quantities of both 3-phase voltage and currents. The model is of course an RCL circuit so no more dynamic equations are needed to make the model more accurate.

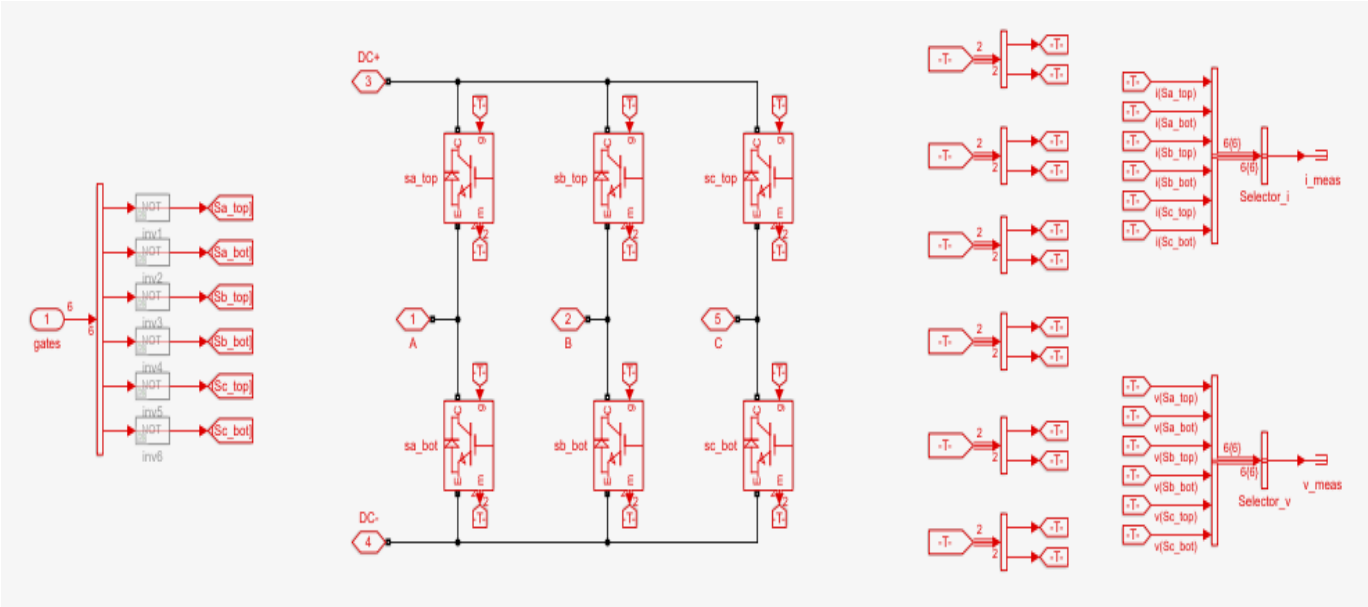

Figure 3.5 Details of 3-phase inverter block in Simulink

The Voltage Source is implemented as an ideal voltage source, this is of course a strong approximation because real batteries also are influenced from different phenomenon as saturations and thermic issues. For future development it could be suggested to implement a dynamic model of the battery, also in the Battery Pack Management System the limitation of the maximum current circulating can be used to pass a variable value of Voltage calculated with the current circulating current value.

Controller

From now on the controller blocks are reported in a specific order, this because it's necessary to introduce one by one the controls once they are intended to be gradually more external loops. The figures used are all been extract from the Typhoon Schematic Editor framework this is just for graphical clarity reasons but the algorithm has firstly been developed in Simulink and then imported and used in Typhoon.

3.2.3. Clarke-Park Transformation

According with the general scheme introduced in Figure 1.36 it's clear that controller and plant use two different types of current frameworks; respectively direct and quadrature frame and a-b-c three phase reference. To convert the 2 frameworks, the model needs a block called "dq to abc" that basically converts the voltages reference Vd* and Vq* into the respective three phase frame Va, Vb and Vc that feed the inverter once they have been passed through the SV-PWM generator that build the PWM signal.

It's interesting to notice that both Simulink and Typhoon PMSM blocks directly provides Id and Iq currents as feedback because the same inverted block is embedded inside the motor's ones, so it's not necessary introduce the second block to re-calculate the direct and quadrature currents.

This "dq to abc" block contains the two transformation matrices known as "Clarke and Park transformation" already described in Chapter 1. Those are mathematical tools the change the benchmark from a 3-phase reference into a 2-phase one.

To be clearer here is resumed one by one each transformation with the relative picture that show the graphical representation of the signal named "abc 3-phase", "αβ 2-phase" "direct and quadrature".

Clarke Transformation

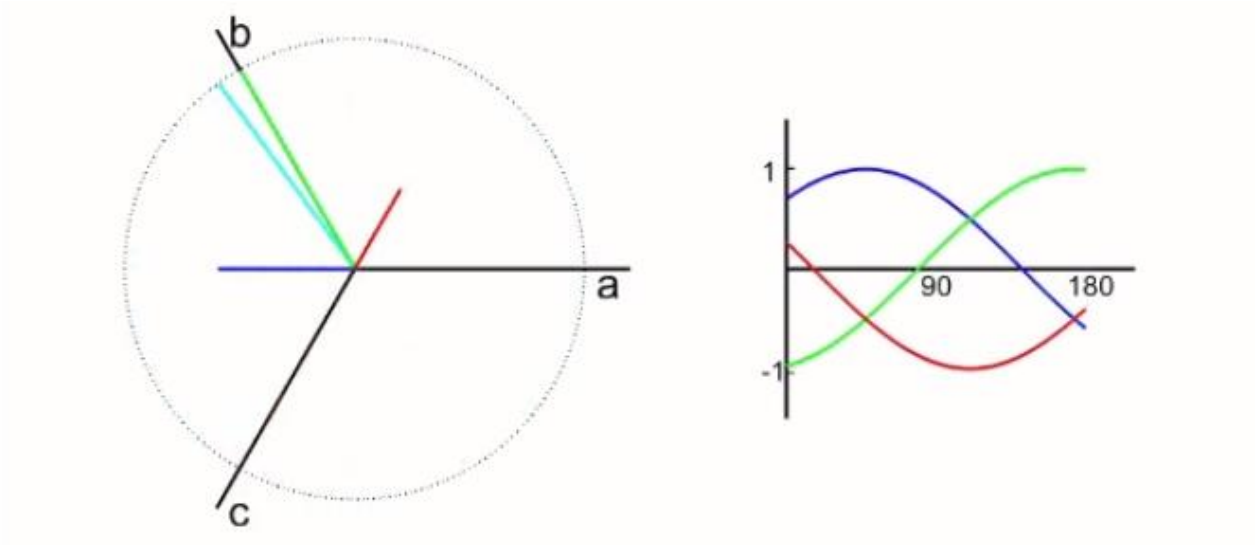

Figure 3.6 3-phase currents reference frame.[41]

In Figure 3.6 is shown the well-known trend of a simple 3-phase signal multiplying for the following matrix is possible to obtain the αβ 2-phase equivalent signal:

$$i_{\alpha\beta o}(t) = P i_{abc}(t) = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}\begin{bmatrix} i_a(t) \\ i_b(t) \\ i_c(t) \end{bmatrix}$$

Which is represented in the following figure that show the trend of an αβ 2-phase signal.

[41] https://it.mathworks.com/solutions/power-electronics-control/clarke-and-park-transforms.html

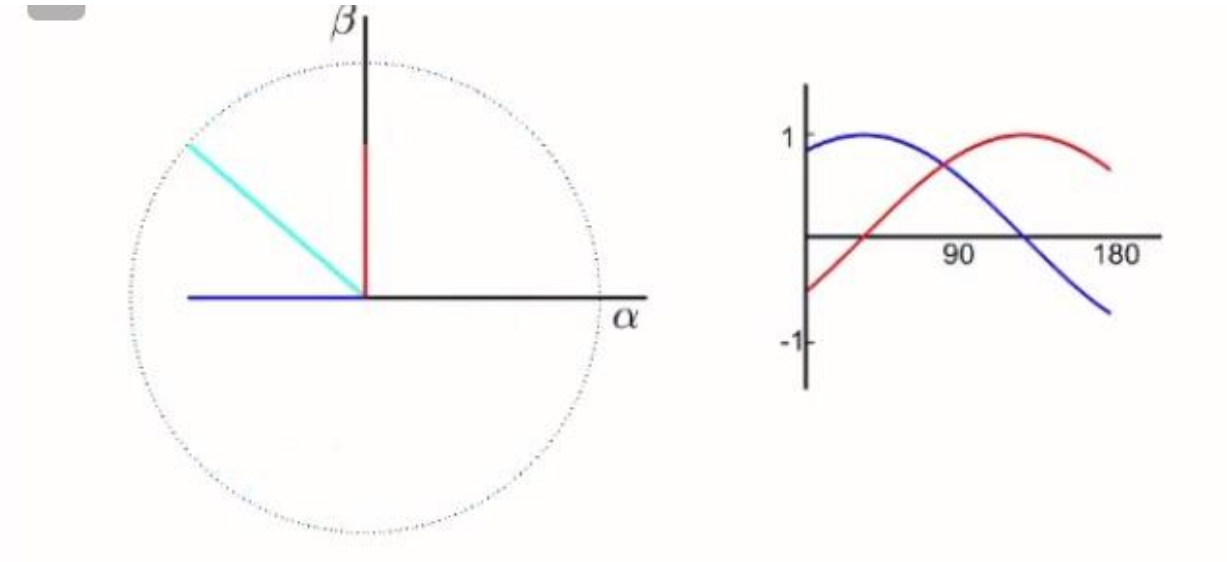

Figure 3.7 α-β 2-phase currents reference frame.

Park Transformation

Once obtained the αβ 2-phase is possible to calculate the equivalent direct and quadrature signal by multiply for the following matrix:

$$K_{CP} = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos(\theta) & \cos\left(\theta - \frac{2\pi}{3}\right) & \cos\left(\theta + \frac{2\pi}{3}\right) \\ -\sin(\theta) & -\sin\left(\theta - \frac{2\pi}{3}\right) & -\sin\left(\theta + \frac{2\pi}{3}\right) \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}$$

It' important to notice that also the angle θ is required, so in the model the feedback θ (angular position) is passed to the "abc to dq" block. The following figure (pic) shows the trend of a direct and quadrature signal.

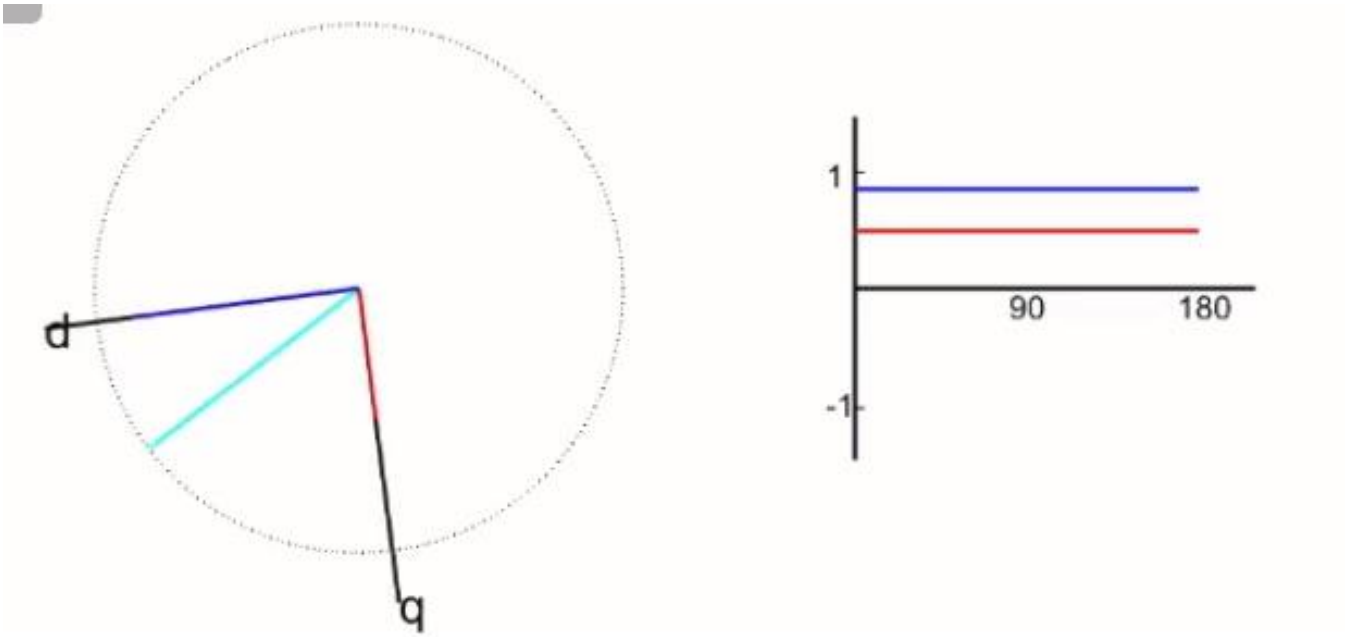

Figure 3.8 Direct and Quadrature currents reference frame.

Since in the model we want to convert the direct and quadrature voltages into the relative abc 3-phase one, it's easy to understand that the "abc to dq" block operates with the inverted matrix of the two transformations. They are following reported.

$$K_{CP}^{-1} = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos(\theta) & -\sin(\theta) & \frac{\sqrt{2}}{2} \\ \cos\left(\theta - \frac{2\pi}{3}\right) & -\sin\left(\theta - \frac{2\pi}{3}\right) & \frac{\sqrt{2}}{2} \\ \cos\left(\theta + \frac{2\pi}{3}\right) & -\sin\left(\theta + \frac{2\pi}{3}\right) & \frac{\sqrt{2}}{2} \end{bmatrix}$$

$$i_{abc}(t) = \frac{3}{2} \begin{bmatrix} \frac{2}{3} & 0 \\ -\frac{1}{3} & \frac{\sqrt{3}}{3} \\ -\frac{1}{3} & -\frac{\sqrt{3}}{3} \end{bmatrix} \begin{bmatrix} i_\alpha(t) \\ i_\beta(t) \end{bmatrix}.$$

3.2.4. Direct and Quadrature Currents

The first control is of course the Vectorial control of the currents in which Vd* and Vq* are computed from the calculated reference currents Id* and Iq* and the feedback value provided form the sensor. Within reference to Figure 1.14 where the equivalent circuit is represented and with reference to the equations showed in Section 1.2.2.2. here reported:

$$\begin{cases} v_d = R_s\, i_d + \dfrac{d\lambda_d}{dt} - \omega_e\, \lambda_q = R_s\, i_d + L_d \dfrac{di_d}{dt} - \omega_e\, L_q\, i_q \\ v_q = R_s\, i_q + \dfrac{d\lambda_q}{dt} - \omega_e\, \lambda_d = R_s\, i_q + L_q \dfrac{di_q}{dt} - \omega_e\, L_d\, i_d + \omega_e\, \lambda_{PM} \end{cases}$$

A saturation block is also inserted at the end of the calculation; this it's necessary since the voltages in the inverter can't in any case be higher than their maximum rated value of 230.9 V remembering that:

$$V_{line} = \sqrt{3} V_{phase}$$

It acts when the references change too quickly and the controller steps into any wind-up issue the final saturation minimize the problem.

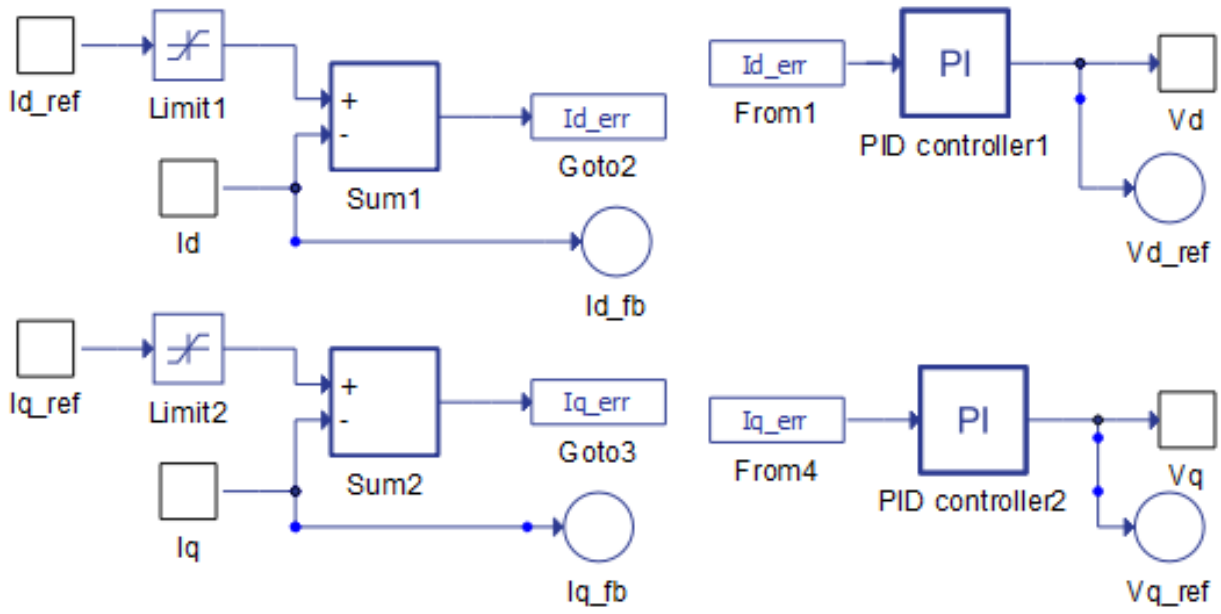

Figure 3.9 This is the subsystem named "V dq calculation". In the left-side Id and Iq error is computed and in the right-side the PI acts to produce Vd and Vq reference.

3.2.5. Torque Control

This is the most complex phase of the controller. Here the Id* and Iq* reference is computed as result; the input otherwise is Torque reference we want to obtain.

In the first part it's easy to notice that a switch block is used to discriminate if the T* should be the one computed in the speed controller or it must be the one passed directly by the user applying the pressure on the acceleration pedal.

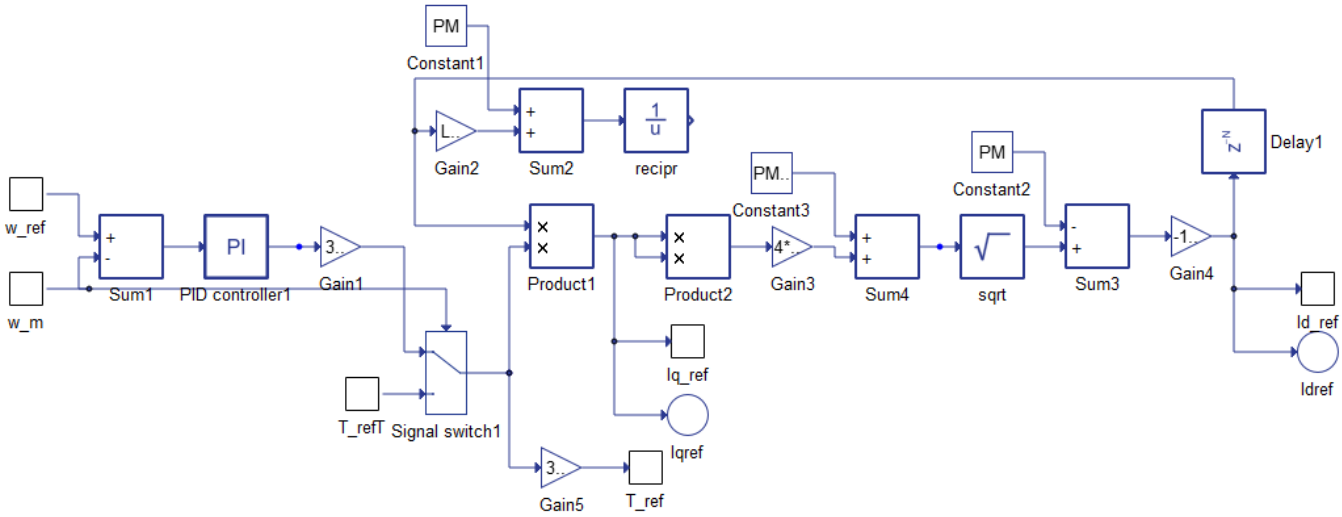

Figure 3.10 This is the subsystem named "Speed Control" here speed error is calculated and controlled with a PI controller than the Id and Iq reference value are computed according with the MTPA equation described.

MTPA Region

The reference Torque is used to calculate the Iq* with the MTPA region equation explained in Chapter 1 and here reported:

$$T_{em} = \frac{3}{2} pp\left[\lambda_{PM} + (L_d - L_q) \cdot i_d\right] i_q$$

Then as in Chapter 1 the Orthogonality relation is used to calculate the Id*.

$$i_{d\_mtpa} = \frac{\lambda_{pm}}{4(L_q - L_d)} - \sqrt{\frac{\lambda_{pm}^2}{16(L_q - L_d)^2} + \frac{I_m^2}{2}}$$

A time delayer is inserted due to avoid a bad algebraic loop in the solver so the current value of Id* is used to calculates the next one. Those reference currents are directly passed to the second phase of the calculation.

Field Weakening Region

As explained in the chapter 1 the field weakening region is reached when the feedback speed reaches the value of "base speed" in other words when the voltage limits (chapter 2) is overcome. At this point the currents must change their values to allow the speed to increase keeping the power to the same value. Here is shown the block named "FW Region"

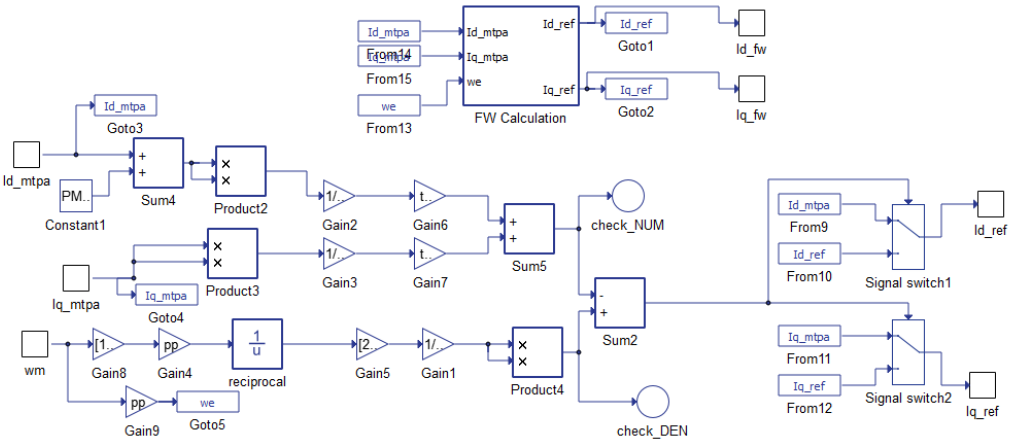

Figure 3.11 This subsystem is the one named "Fw region". Here FW condition are verified and the right Id and Iq couple of current is selected.

The main thing to notice is the voltage limit calculation the enable the block "FW Calculation" once the limit is reached. In this case it's been necessary to introduce also a special "gain" to reduce the memory needed from the CPU because the values named "NUM and DEN check" are too small quantities and they cause the Overflow of the CPU.

At the end other two switches are activated to select the correct signal to use in the two cases of "MTPA" or "Field-Weakening" condition previously computed. It's obvious to notice that "Field-Weakening" condition is "NaN" till the block is enabled.

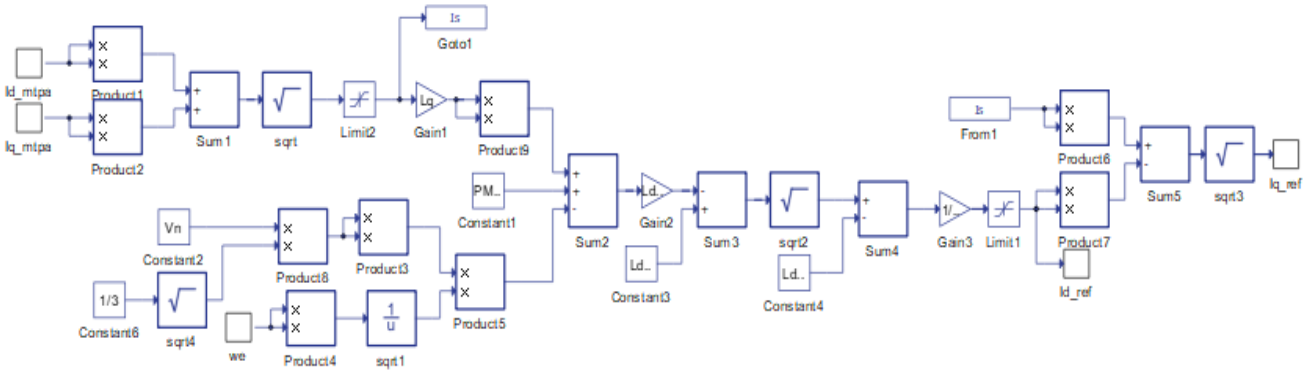

Figure 3.12 This is the subsystem inside "FW region" subsystem (Figure 3.11). Here, when opportune, Id and Iq in the Filed Weakening region are computed according with the relative equation discussed and passed to the selector in the previously subsystem.

In Figure 3.12 is reported the entire calculation of Id* and Iq* in the field weakening case. As is explained in chapter 1 that calculation of Id* is done using the following equation:

$$i_d^* = \frac{-2L_d + \sqrt{4L_d^2 - 4(L_d^2 - L_q^2)\left(1 + L_q^2 I_{max}^2 - \frac{V_{max}^2}{\omega^2}\right)}}{2(L_d^2 - L_q^2)}$$

Iq* is consequently computed as:

$$i_q = \sqrt{I_{max}^2 - {i_d^*}^2}$$

3.2.6. Speed Controller

As it's mentioned in the MTPA region paragraph the T* can be computed from the speed reference, this is an external not necessary loop of the controller that characterizes a specific driving condition in which is the priority is the regulation of the cruise speed (cruise control) so the reference speed is imposed by the user and the PI controller elaborates a reference Torque value in order to achieve the exactly reference speed.

The PI control has also an anti-wind-up system integrated because it could be very common that the reference is far from the actual speed and so the gains could determinates wind-up issues. The technique used in this case is a simple "clamping" of the integrator action ones the output value reach the maximum torque imposed (that in this case is 2/3 of the maximum torque achievable according with the producer).

3.2.7. Battery Pack Management System

Lastly, a BMS is implemented to simulate the real behavior of a real battery pack (even though the source of voltage simulated is still an ideal one). The BMS acts directly to the saturation value imposed in term of "slope" that of course is directly dependent on the maximum torque achievable, as we just mentioned in the last paragraph.

This block receives as inputs the feedback currents Id and Iq but also the voltages Vq and Vd after the saturation in the Vectoral controller. So, a rms value of current is computed which represents the currents circulating in the power electronic components of the plant.

$$\hat{I}_{DC} = 3\frac{V_{RMS,phase}I_{RMS,phase}}{V_{DC}}\cos(\varphi) = \frac{3}{2}\frac{V_{pk,phase}\,I_{pk,phase}}{V_{DC}}\cos(\varphi)$$

This current value is compared with a specific limit that is set by the user according with the safety criteria that is needed (so it will be a certain % of the maximum current available before thermal issue occurs in the specific duty type).

If the running current overcome the imposed value a dynamic variation of the maximum slope is passed as a slope reference. It's important to notice that this value depends directly on the "error" value and its determinates the same percentage of variation in the slope. As it's easily shown in figure (pic) the maximum value of currents can be positive (charging) or negative (discharging), for many different reasons it' also possible to set 2 different values i.e. according with the type of duty.

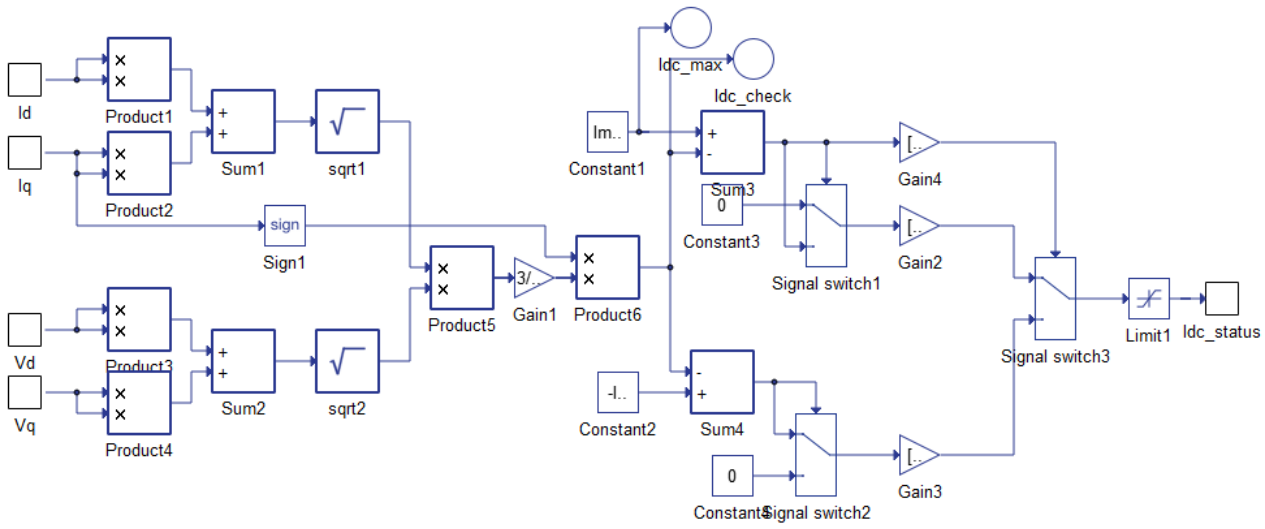

Figure 3.13 This subsystem (Battery Pack Management System) provides the maximum currents allowed in the plant.

3.2.8. User Input interface

Since now it has been presented how the algorithm works, but we miss one important point yet and this is how the user can interact with the algorithm and change the input signals to see how the outputs change.

To do so in Typhoon we use the SCADA panel. This panel has lot of functionality but for this work what is most important to introduce are the input and visualization widgets. In Figure 3.14 is shown a screenshot of what it's been set in the SCADA panel: from left side to the right are visible the input widgets:

- Inverter enable that shouts down the inverter when disactivated;
- The Load Torque: that will not be modified in this work and keep constant, but it represents the external disturbance to the system as the wind or the friction effect;
- The Torque demanded: In other words, the pressure on the accelerator pedal transduced into a signal. Its value can be changed by the user between 0.12 (the minim torque that could win the static forces and let the system move) and 1 (which represents the maximum value of torque available without any risk, to be clear this is exactly 2/3 of the maximum Torque as it's been mentioned more than one time in this chapter);
- The speed reference: Expressed in rpm that means the speed the driver want to reach and maintain (cruise speed).
- The other widgets are just a few measured output signals that are helpful in a 1st evaluation of the simulation; much more important is the Capture/Scope widget where all the signals can be visualized, saved, and transferred into a MATLAB workspace in order to be compared with the Simulink model simulations.

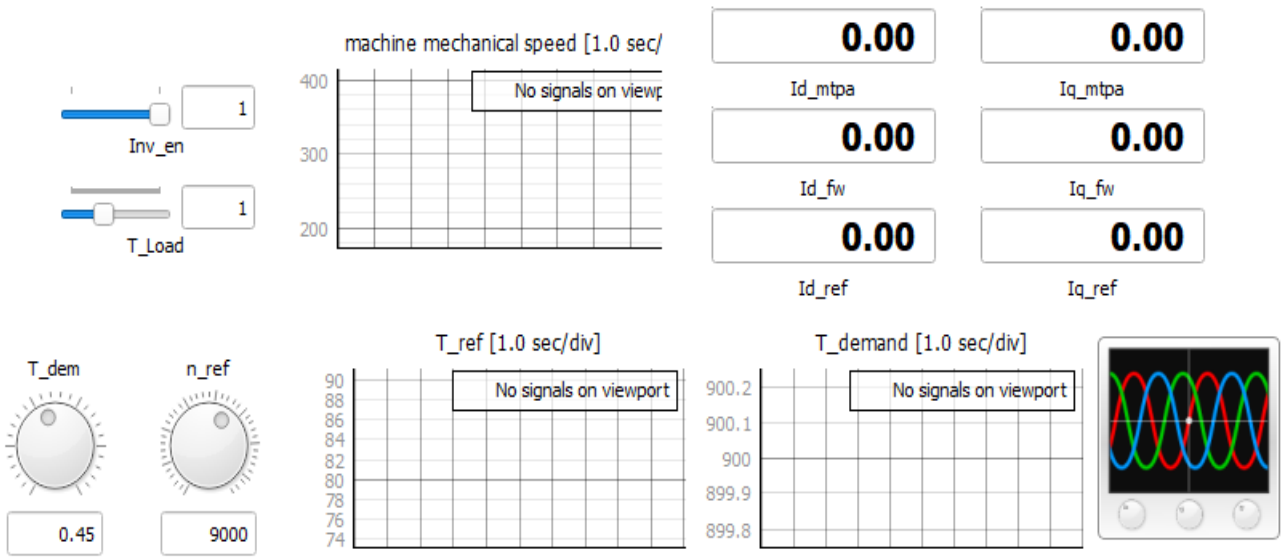

Figure 3.14 SCADA User interface, on the left side the 4 Input by the User. The others are just a sample of widgets. On the bottom right, the Capture/Scope block which let possible to save the data acquisition and upload them on MATLAB script (see Appendix).

Figure 3.15 shows the subsystem named "User Input interface" here is clear how the acts work on the algorithm:

- The speed reference is converted into "rad/s" and passed to the Rate Limiter Dynamic;
- The Torque demand is converted form a %value into a "Nm" value and passed to the MTPA subsystem when it becomes the T* reference the allows the currents reference calculation.

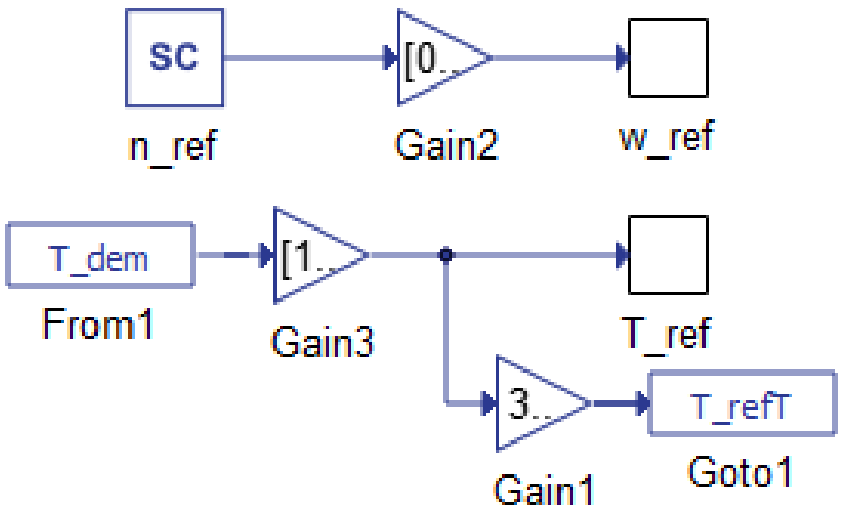

Figure 3.15 User Input signal from SCADA

Look Up Table for transition speed reference computation

One last thing to notice is that when the system works without speed control (T* is directly passed to the controller MTPA section) and it suddenly must be switched into the speed control or vice versa the reference speed passed as 1st value must be congruent with respect the final value reached in the previously mode. This is necessary to the controller because if it's not respected the speed error could increase a lot and generates a full saturated signal the basically could stuck all the algorithm. To overcome this issue without creating other heavy variables or function that could make the computation more difficult it's been inserted a special look-up-table that according with all the parameter of the specific simulation generates a "virtual" value of slope that determinates the "correct" value of speed reference once the system switch its control domain.

The Look Up Table is always the best choice in terms of computation but of course there also many limits due to the "empirical" value used because they are strongly depended on the CPU performance itself and so they must be settled for each system. One other aspect is the limitation in term of operating borders and number of points included in the linearization process that must be done during the development phase.

In this work just Torque demand value has been considered in the 1D Look Up Table but of course also the T load should be inserted in the computation, in this case it's not necessary because it's been chosen a constant value.

The switching speed reference also has been used as a constant but if we want to be more accurate the speed should be selected by the user too and included in the LUT definition; so, including all the effect the LUT should be a 3D LUT.

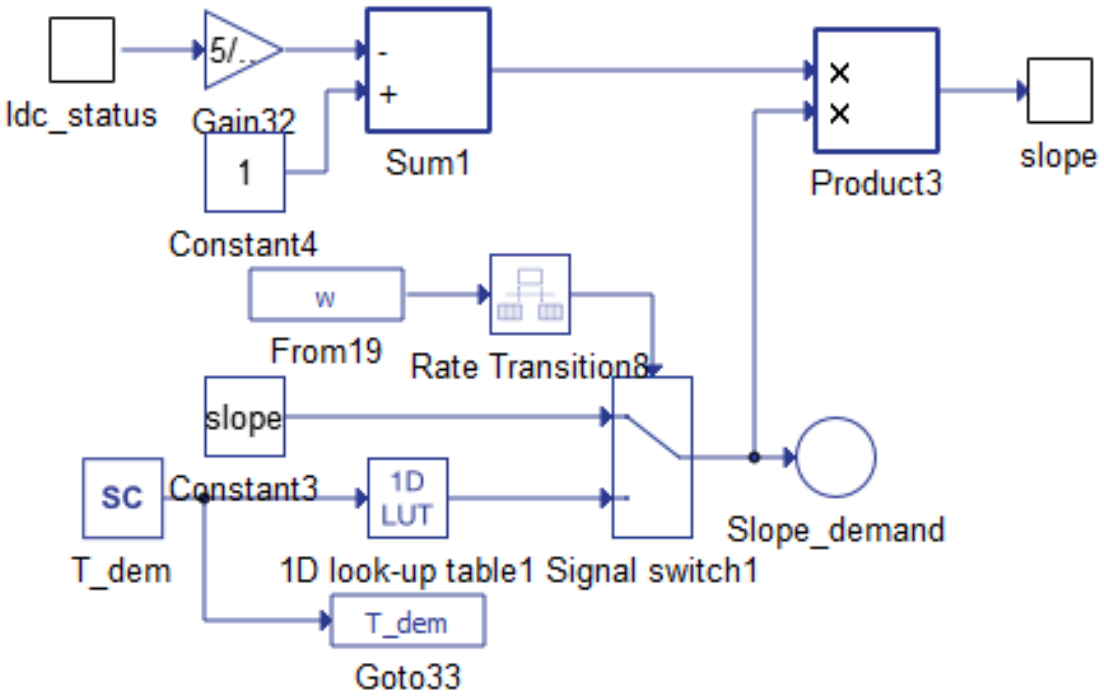

Figure 3.16 In the upper part is shown the percentage regulation of the I_dc where a (0-1) number is calculated. In the bottom's one the slope is computed and modified if the BMS is working.

Finally, is presented a list of the Simulation parameter used for the Simulation:

- The PI's parameter has been chosen with a trial-and-error system.
- The Step time's and their ratio has been chosen as a compromise between time of computation and accuracy of the dynamic response;
- The switching frequency of the inverter has been set as 10 times higher than the frequency of computation according with the Nyquist criteria.

Parameter	Symbol	Value
Plant time step	Ts	2 [μs]
Controller time step	Tsc	40 [μs]
Inverter switching frequency	fsw	50 [kHz]
Proportional gain Id_calculation	Kp_id	1 [Ω]
Integral gain Id_calculation	Ki_id	300 [Ωs]
Proportional gain Iq_calculation	Kp_iq	11 [Ω]
Integral gain Iq_calculation	Ki_iq	6500 [Ωs]
Proportional gain speed controller	Kp_w	5 [Nms/rad]
Integral gain speed controller	Ki_w	80 [Nms^2/rad^2]

Table 1 All the simulation parameter used in the simulation and the controller parameters adopted.

4. Simulation and Results

4.1. Simulation Classifications and Data Collecting

In this last Chapter the focus will be on the results achieved during the testing phases. Moreover, the test bench building will be discussed in all the intermediate steps gained. At the end of this Chapter, also, the limitations of the test bench will be taken into consideration.

The process followed to build the test bench starting from the theory presented in Chapter 1 can shortly be described in this list of steps:

- Build the Algorithm in Simulink and test with an easy MATLAB script if the system's behavior is exactly like the literature's consideration suggest: Starting from an Open-Loop system and tuning one-by-one all the controller's parameters to make the system behavior to be the less instable as possible;
- Convert the entire algorithm into a .Tse File and export it into Typhoon Schematic editor; Once the simple modifications introduce in Chapter 2.1.1. have been completed test the system in with an easy SCADA Panel and collect the simulations outputs with the auxilium of Capture-Scope tool;
- Collect all the control unit part of the system in one single block and let Typhoon to generate the C-Code; Once the Code is generated following the procedure illustrated in Chapter 2.2.3 in MATLAB create your S-Function block and substitute the control-unit part with it;
- Build a MATLAB script to make comparison of the 3 Virtual Simulations;
- Now, introducing the Real-Time simulation as first step: Do operative considerations about the clock-time required by the system and his compatibility with the performance of the hardware available; At the very beginning just simulate the same system switching into the RT-Mode and modify the two clock-time available (the plant and the controller) in order to understand which can be the bottle-neck;
- Now introducing also the Typhoon 602+ embedded breakout board test the real HIL architecture using also the physical signal as is detailed in Chapter 4.3.1;
- Using the LabVIEW environment to pilot the SPARK ECU follows the procedure described all along Chapter 2.3. and close again the loop between Typhoon and SPARK breakout boards guaranteeing to the Test Bench the possibility to use a RT OS instead of a common CPU

available in your host PC; it can be necessary to use two different host PCs to run simultaneously the LabVIEW RT Module and the Typhoon HIL software;

4.2. Virtual Time Simulations: Typhoon – Simulink Comparison

This paragraph presents a sample of significative comparison between the two models, as mentioned before the simulations have been conducted with many different parameters in term of simulation parameters, model characterization and user requirements.
Before coming into the main part of the analysis in Figure 4.1-5 are reported an entire plot of a single Simulation (#2 in the Table) and each graph is better explained:

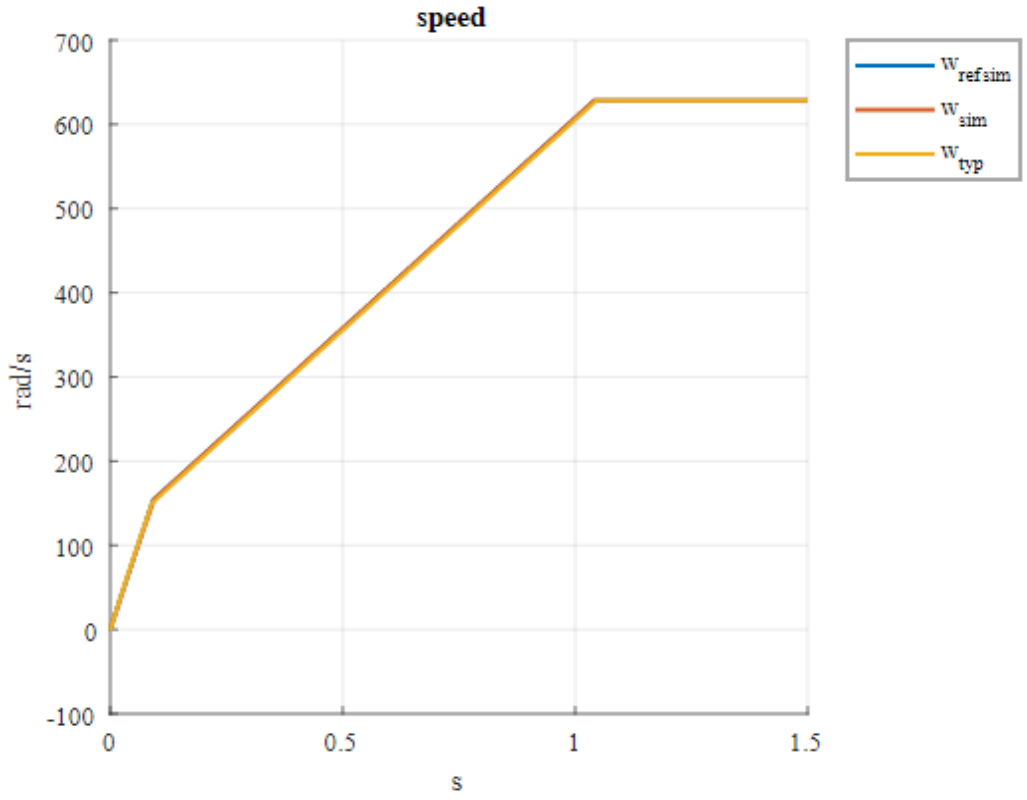

Figure 4.1 Comparison between speed in Simulink and Typhoon (Simulation #2).

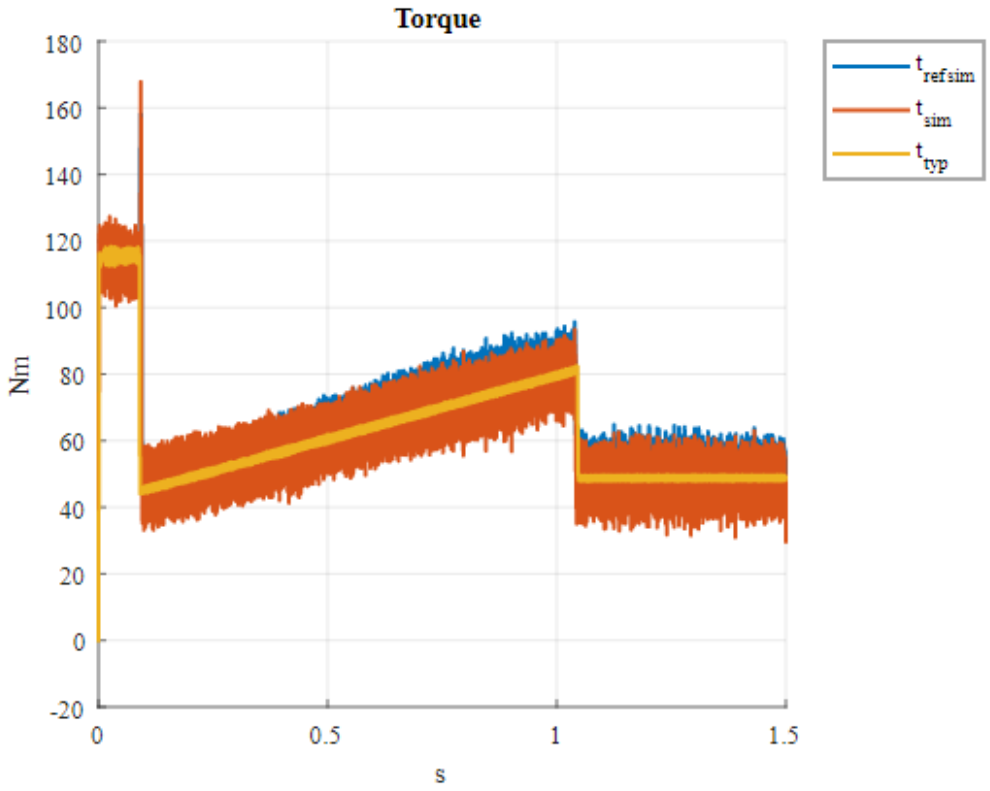

Figure 4.2 Comparison between electric torque in Simulink and Typhoon (Simulation #2).

In these first 2 pictures the comparison is made between all 3 version of the Simulations. The speed reference passed is 6000 rpm that in the graph corresponds around 628 rad/s. The Torque is 100% of the nominal so basically 50% of the maximum one around 118,5 Nm.

What is possible to get from these first 2 picture is now reported in pills:

- There is a certain consistence in the simulation results;
- Despite of the same filter applied Typhoon's Simulations results have very much noise less; this can be attributed to a weakness of the sensor's model used by the MATLAB simscape electric libraries contained inside the Typhoon HIL conversion libraries;
- From the electric torque plot it's also well visible a certain imprecision of the control that creates a peak in the Simulink versions of the simulations in correspondence of the switching status between direct torque and constant acceleration. Even if the peak doesn't reach its saturation value (Maximum Torque 237 Nm, at least in this case) there is a clear issue with the optimization of the Look-Up Table, indeed it's been developed directly in the Typhoon model (because of reasonable questions of timing: the SCADA Panel allows to change

parameters in the middle of the simulation without any re-compiling needs), so, there can be two main different reason to this:

1. The error the Occurs is exactly the same and the Look Up Table is completely compatible between the two versions but the noise disturbance create this bigger amplification in the Simulink cases.
2. The error is due to the Look Up Table it-self, so it needs to be re-build in Simulink minimizing the error. Of course, this can't be done proficiency without the using of a properly fitting tool.

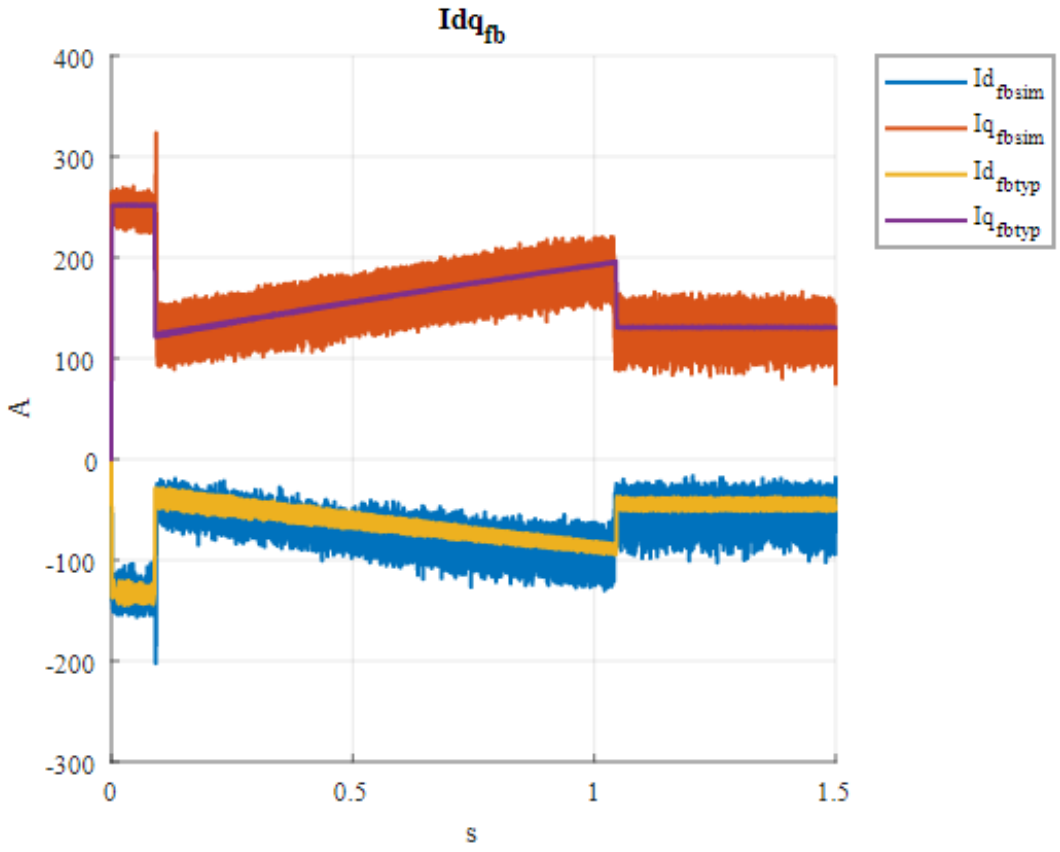

Figure 4.3 Comparison between i_d and i_q currents in Simulink and Typhoon (Simulation #2).

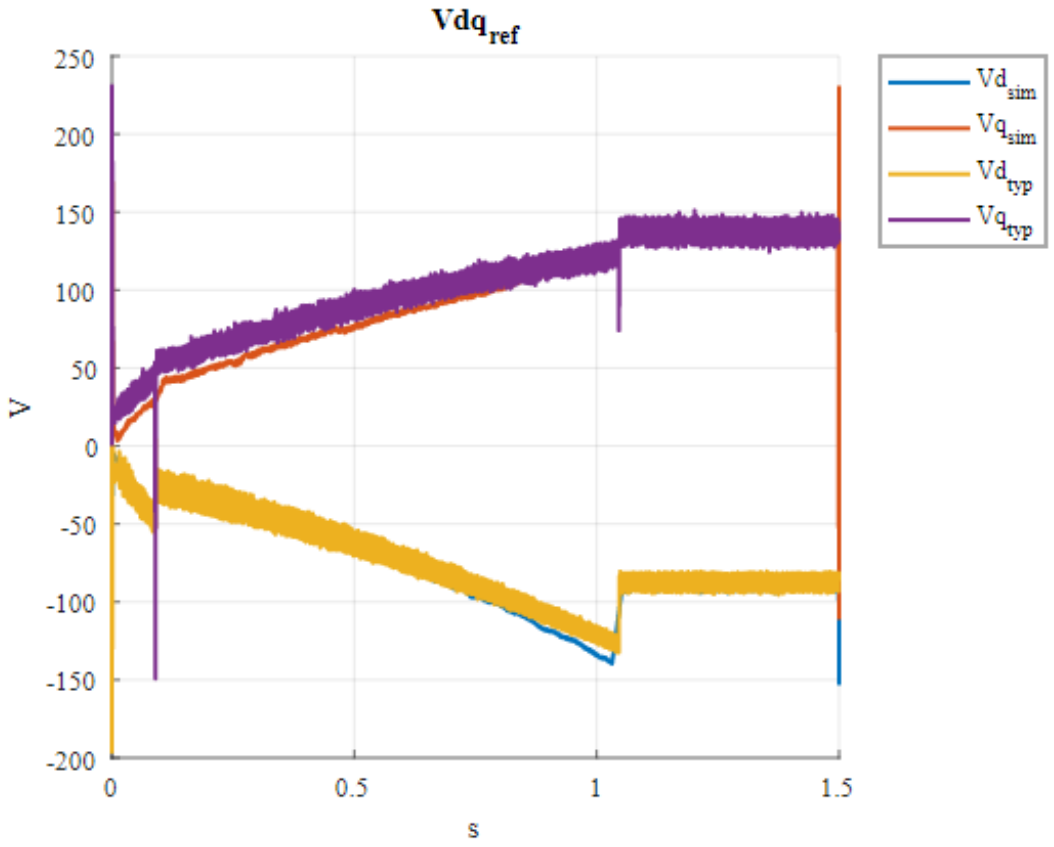

Figure 4.4 Comparison between V_d and V_q Simulink and Typhoon (Simulation #2).

On these other two pictures just one case of Simulink has been plotted, this just because the figure could be less readable with more two signals. But in any case, exactly as it's possible to see from the previously two pictures there is no substantial difference between the Simulink's cases (the original one has been neglected. What is possible to say about these is reported in pills:

- All the considerations already done can be extendable;
- About the currents pictures it's clearly possible to notice that also in the Typhoon's simulation it's possible to observe a certain amplification of the noise between i_q and i_d. This is a good confirm of the solidity of the consideration done in the Theory part. In particular the direct current is calculated from the quadrature one that is obtained directly from the Torque reference (see MTPA paragraph) so it's necessary that the error is amplified;
- In the Voltage graph is possible to notice that the noise present in the Simulink case is much less than the Typhoon's one. This can be justified by the nature of this computation: indeed this is not consequence of a measure by the sensors but is done mathematically from the CPU. So, basically the difference stands on the SV-PWM block used that's different in the two versions (see .TSE conversion paragraph).

As already mentioned this first example is done by the Simulation #2 on the table reported in the bottom of the paragraph. But there are some other aspects to take into consideration about the typology of Simulation done:

4.2.1. Flux-Weakening condition Analysis

- In Figure 4.5 is reported the i_d-i_q reference frame where all the recorded value has been plotted. Also, in evidence there are the voltage and current limits explained in Chapter 1. It's clear that no "Flux-Weakening" conditions are gained. Indeed, the minimum speed to gain the "Flux-Weakening" conditions is around 720 rad/s. In the Simulink case it's clear that some of the points seems to overcome the Voltage limits, but this is simply justified with the consideration about the high noise done in the first comment.

- In Figure 4.6 a simulation that include the Field-Weakening condition is shown. The picture clearly shows the voltage limit (hyperbole) in green reached by the currents couples.

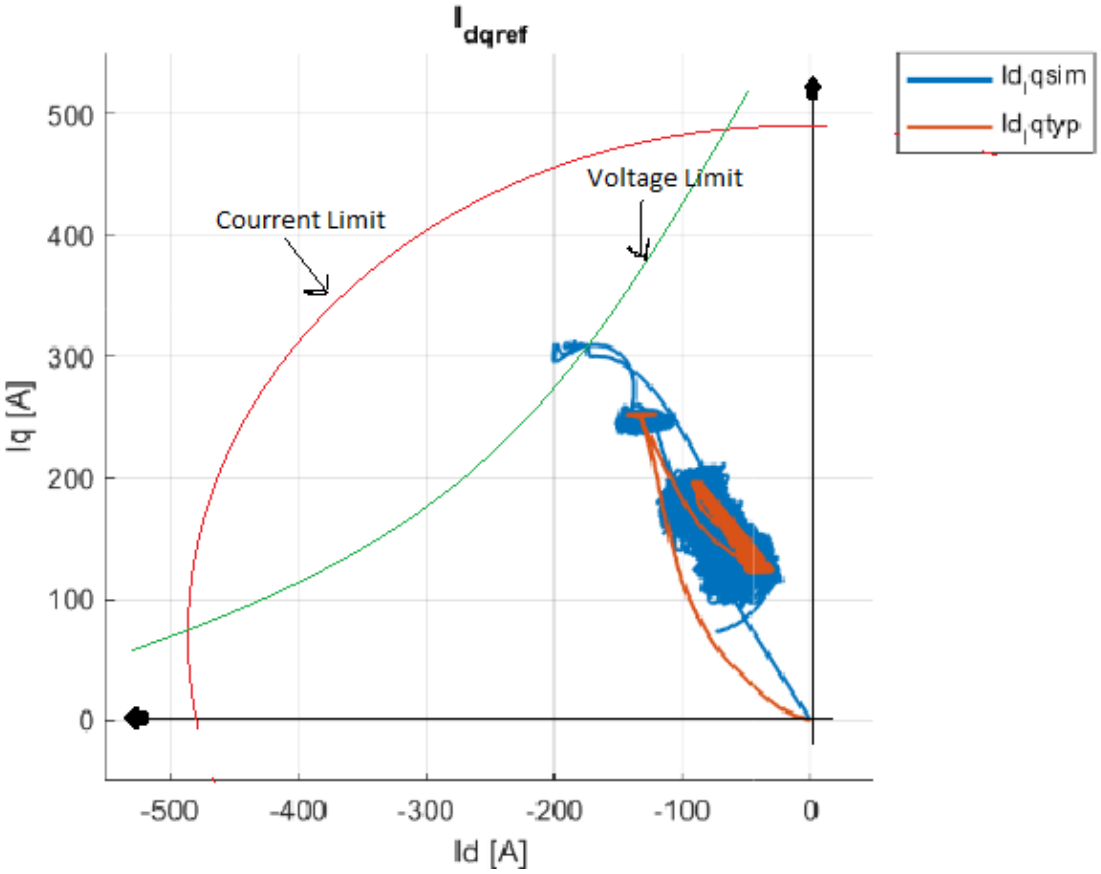

Figure 4.5 Couples of i_d and i_q in the d-q reference in Simulink and Typhoon (Simulation #2). The green hyperbole must be considered a qualitative representation.

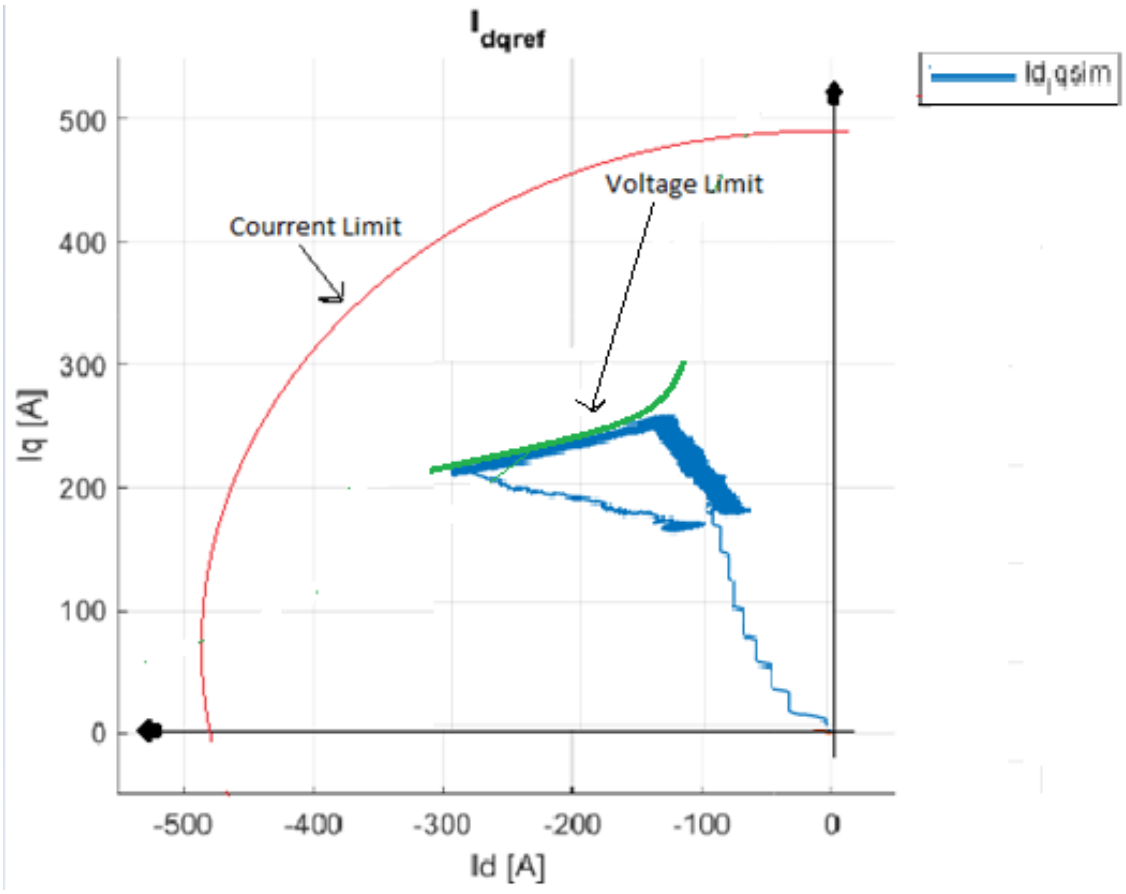

Figure 4.6 Couples of i_d and i_q in the d-q reference only Typhoon Simulation (Simulation #27).

A couple of other simulations are now exploited:

Case A) This simulation done without changing the control from Torque to Speed Controller (See User Input Interface paragraph): in this case the PI control done in the Speed-Controller phase is neglected and only the Torque reference is directly passed to the next step: MTPA calculation of direct and quadrature currents.

It's clear that in this case no linear growth of speed is executed but a quadratic evolution is expected (see paragraph 1.5.)

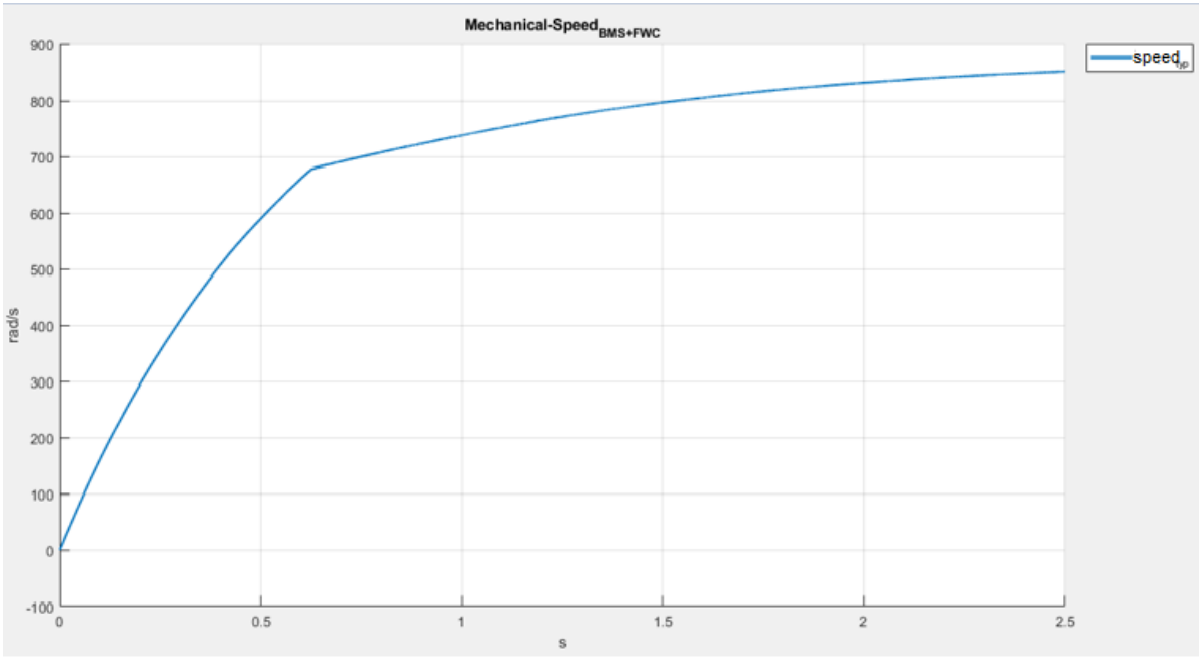

Figure 4.7 Speed evolution in the case A. In this simulation also the BMS condition are reached.

Case B) This simulation is not referred to any of the cases present on the table. This time, through a simple command, both speed and torque reference are settled to zero after 1 second of simulation.

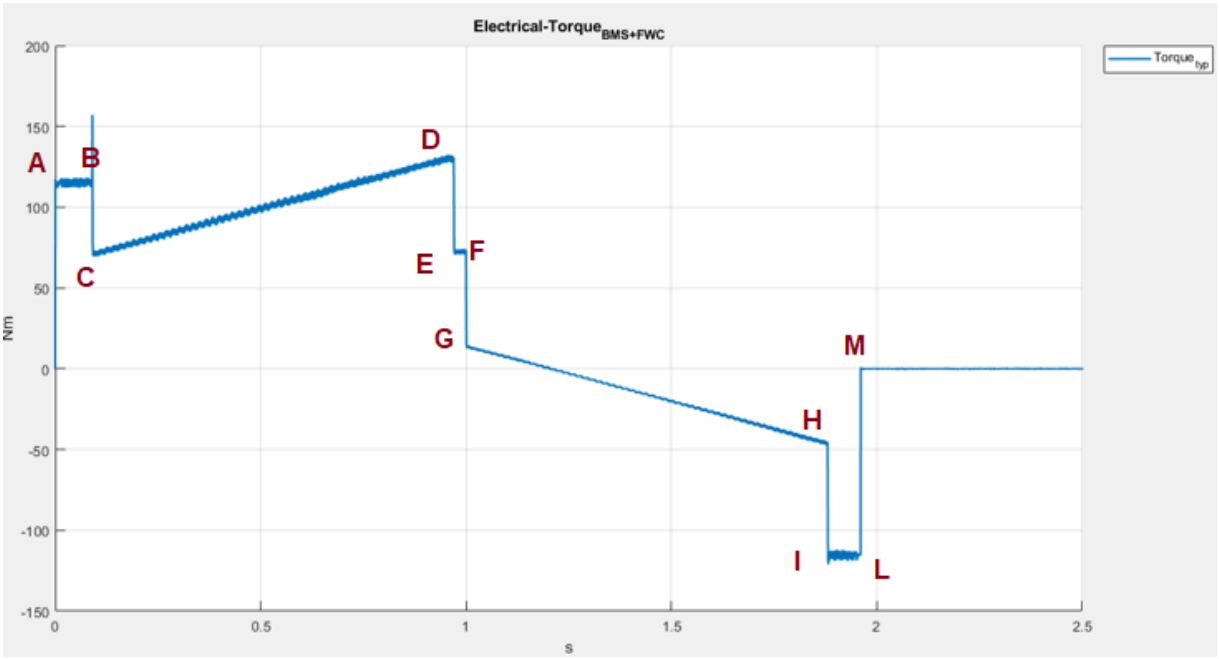

Figure 4.8 Torque evolution in the case B.) In this simulation after 1 second of simulation both torque and speed reference are settled to zero.

In

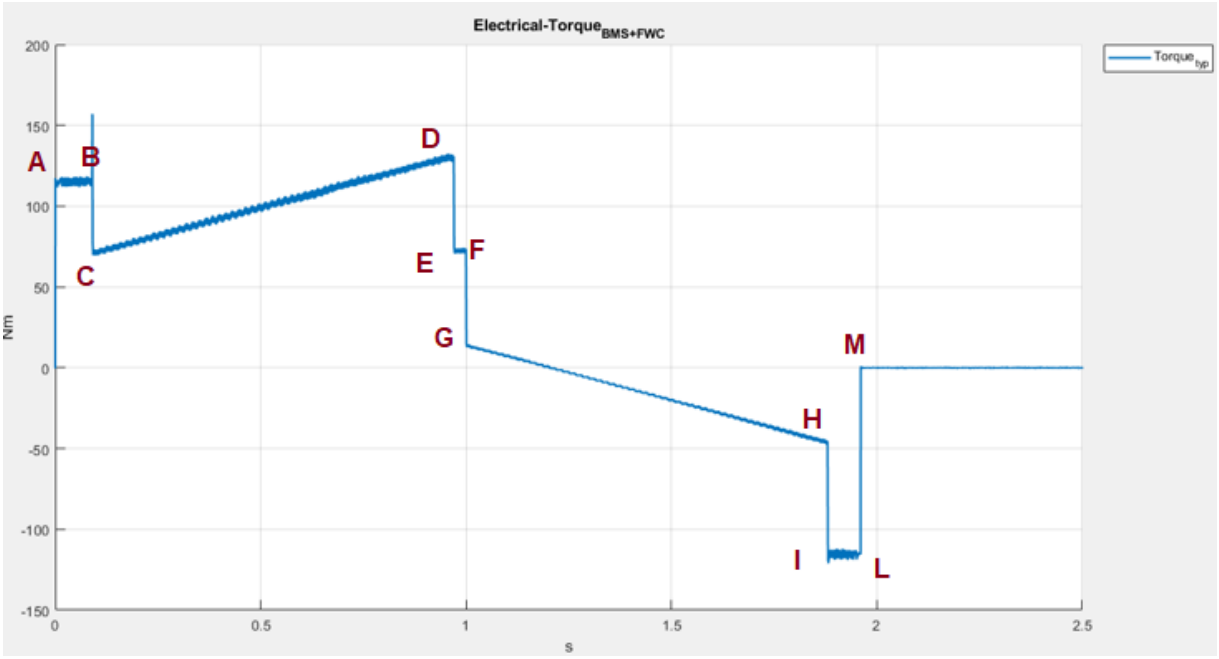
Electrical-Torque BMS+FWC
Torque typ
Nm
s
A
B
C
D
E
F
G
H
I
L
M
200
150
100
50
0
-50
-100
-150
0
0.5
1
1.5
2
2.5

Figure 4.8 The Electrical-Torque feedback is shown: Here what happens to the system can be summarized in this way:

A-B) The Nominal Torque is provided to the controller (the Speed Control is disabled till 150 rad/s).

B-C) 150 rad/s is reached and the control typology is switched on the speed control (the peak is justified by the instability already discussed about the LUT analysis).

C-D) The nominal evolution of speed is shown. Point D represents the speed target is reached.

D-E) The Torque is decreased because the dumping component of the couple equation is shut down.

$$T_{em} = T_{laod} + B\omega_m + J\dot{\omega_m}$$

E-F) The Torque is kept constant because nominal speed is gained.

F-G) The signal is now brought to zero again and the torque decrease again, but this time to slow down the motor.

G-H) The Torque is now increasing (but with a negative value) till the speed of 150 rad/s is reached again (point H).

H-I) As 150 rad/s is reached the step is again visible due to the changed condition of control.

I-L) Nominal Torque is again settled (now with a negative value) to slow down the motor till 0 rad/s.

L-M) Now the Motor is completely stopped, all the components are zero e nothing happens till the end of the simulation.

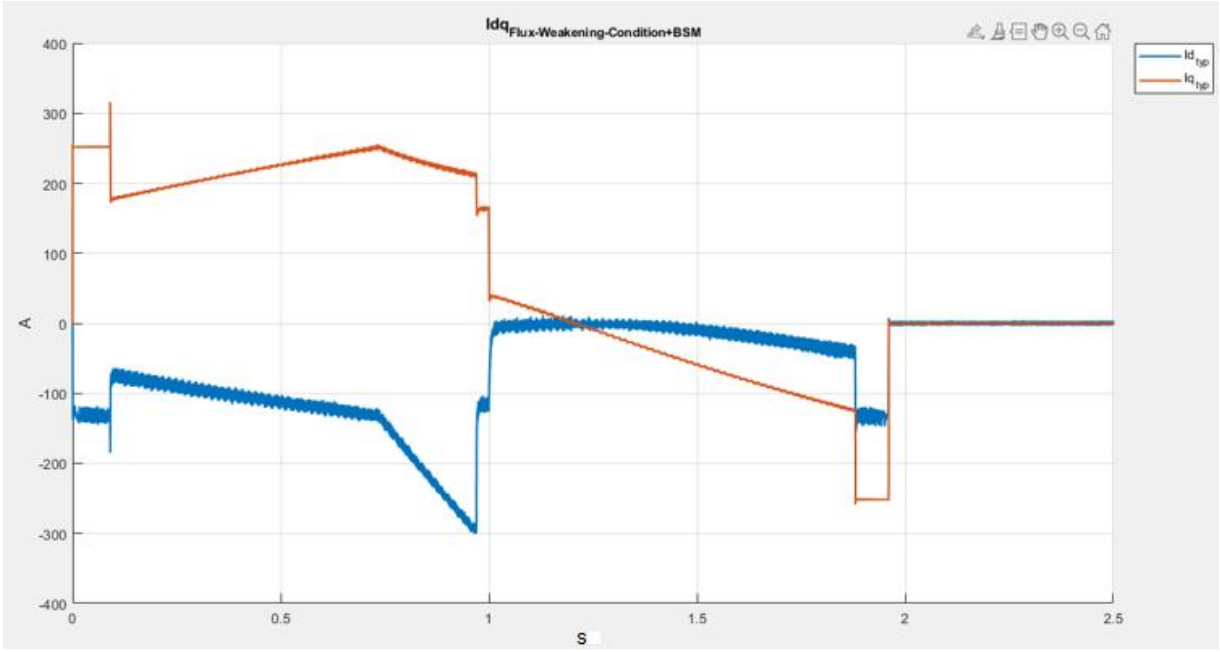

Figure 4.9 Direct-Quadrature currents evolution in the case B.) In this simulation after 1 second of simulation both torque and speed reference are settled to zero.

Again, the summarized situation is illustrated step by step referring to the current's behavior:

A-B) Referring to the nominal Torque the currents are nominal too.

B-C) see torque case.

C-D) Here, is clear that on a certain point the currents have a discontinuity where the Flux-Weakening region starts: Iq is decreased, and Id is increased (by negative value) as is shown also in Figure 4.6.

D-E) see torque case.

E-F) Currents are kept constant because nominal speed is gained.

F-G) Here, again the Iq become negative because the speed is decreasing (see chapter 1.5.1.).

G-M) see torque case.

4.2.2. Automatized MATLAB Script and Simulation's List

It's now reported the MATLAB code used to run the virtual simulations and make the comparison between the 3 Virtual systems. The code will be shortly described in different pieces, but in any case, the script is also fully commented to be easily modified for further kind of simulations/comparisons:

```
clear all
close all
clc

N = 100; % Lowpass_filter_parametre
switching_set_speed = 150; % speed treshold that discriminate the mode of control between Speed and Torque
breakpoints = [0.12;0.15;0.2;0.25;0.3;0.35;0.4;0.45;0.5;0.55;0.6;0.65;0.7;0.75;0.8;0.85;0.9;0.95;1];
table_data = [80.5;146.5;242;334;424.5;514.5;604.5;693.5;783;872.5;961.5;1050.5;1140;1229;1318;1407;1496;1584.5;1673];
cd ..
cd ..
filenames = dir("typhoon_thesis\export_SCADA\VTM*"); % get all the file inside export_SCADA that begins with VTM (Complete)
% This is an example of name to better understand the format:
% CPL_gamma_0.076_6000_rpm_slope_900_max_500_A_100_%.mat
```

In these first lines the fixed parameters are settled, the Look Up table data are defined, the saved Typhoon simulation's output are opened.

```
for i = 1:length(filenames)
    % name(i,:) = filenames(i).name; % name of the file, also name of the cartel inside "figure"
    file(i) = load(fullfile(filenames(i).folder, filenames(i).name)); % Upload the file with all the Data
    split(i,:) = strsplit(filenames(i).name,'_'); % separate the file name so it's possible to get all the variables values
    B(i) = str2num(cell2mat(split(i,3))); % reading B(i) value %Nms
    wref(i) = str2num(cell2mat(split(i,4))); % reading wref(i) value %rpm
    w_ref(i) = wref(i);
    slope(i) = str2num(cell2mat(split(i,7))); % reading slope(i) value %rad/s^2
    Imax_ch(i) = str2num(cell2mat(split(i,9))); % reading Imax_ch(i) value %A%
    T_dem(i) = 1/100*str2num(cell2mat(split(i,11))); %reading T_dem(%) %Nm
```

```
% Simulation Settings
time_step = 1.0; % s
Ts = 0.5*10^-5; % s
Tsc = 12*Ts; % s
dt = 1*10^-6; % s
fsw = 1/(10*Ts); % Hz
Td(i) = length(file(i).time_line)*10^-5; % Time duration used in each Simulink simulation

% Motor Parametres
Vdc = 400; % V
T_max = 237; % Nm
J = 0.06502; % Kg*m^2
Rs = 0.0085; % Ohm
Lq = 215*10^-6; % H
Ld = 86*10^-6; % H
pp = 5; % pole pairs
PM = 0.044; % Vs-Wb permanent magnet flux
PMfw = PM; % Vs-Wb
cosphi = 0.94; %  power Factor
T_res = 1; % Nm
```

In this second part: for each of the simulation reported the variables parameters are settled according to the specific name proposed, also the duration-time of the simulation is taken from the number of lines present in the original data; the other physical parameters are settled.

```
%Simulation
simOut(i) = sim(['LUT_FOC_PMSM_CU3']);

w_sim(:,i) = smooth(simOut(i).w.Data(:,1),N);
 t_sim(:,i) = smooth(simOut(i).Trq.Data(:,1),N);

Vd_sim(:,i) = smooth(simOut(i).Vdq_ref.Data(:,1),10*N);
Vq_sim(:,i) = smooth(simOut(i).Vdq_ref.Data(:,2),10*N);
```

```
Id_sim_fb(:,i) = smooth(simOut(i).Idq_fb.Data(:,1),10*N);
Iq_sim_fb(:,i) = smooth(simOut(i).Idq_fb.Data(:,2),10*N);

%Read Typhoon and Plot results

w_typ(:,i) = file(i).channels_data.w;
t_typ(:,i) = file(i).channels_data.machineelectricaltorque;

Vd_typ(:,i) = file(i).channels_data.ControlUnit_Vdqcalculation_Vd_ref;
Vq_typ(:,i) = file(i).channels_data.ControlUnit_Vdqcalculation_Vq_ref;

Id_fb_typ(:,i) = file(i).channels_data.i_ds;
Iq_fb_typ(:,i) = file(i).channels_data.i_qs;

%Idc_status(:,i) = smooth(file(i).channels_data.ControlUnit_BatteryPackController_Idc_st,2*N);
```

Now the first MATLAB simulation is run, and the outputs are extracted and filtered. Also, the Typhoon data are collected inside they correspond vectors (They has already been filtered before the export). In this case the “LUT_FOC_PMSM_CU3” scripts indicates the S-Function Simulink Script.

```
i

end

i = 0; %re set i to 0. You can't change the index becouse both 5 e CU simulink need "i" as index

for i = 1:length(filenames)
    % name(i,:) = filenames(i).name; % name of the file, also name of the cartel inside "figure"

    %%... see the first loop ....%%
i
end
```

Now the second loop is run, and the second Simulink Script is used “LUT_FOC_PMSM_5F” and this is related to the original Simulink Script without the S-Function. After the two simulations are completed (No need to re-collect the Typhoon data twice, of course) the Plot phase starts. The following part of the script simply create and safe each figure making comparison between the different simulations:

```
% plot

for k = 1:length(filenames)

figure(4*(k-1)+1)
hold on
grid on
plot(simOut(k).w.Time,w_sim(:,k),'lineWidth',1.5) %w_sim_sfunct
plot(simOut2(k).w.Time,w_sim_n(:,k),'lineWidth',1.5) %w_sim_orginal
plot(file(k).time_line,w_typ(:,k),'lineWidth',1.5) %w_typ
legend('w_{sim}_{ref}','w_{sim}','w_{typ}','location','northeastoutside')
title('speed')
xlabel('s')
ylabel('rad/s')
cd figure
cd Plot_double_sim
    cd(filenames(k).name)
    saveas (figure(4*(k-1)+1),['speed.svg'])
    saveas (figure(4*(k-1)+1),['speed.mfig'])
cd ..
cd ..
cd ..

figure(4*(k-1)+2)
hold on
grid on
plot(simOut(k).Trq.Time,t_sim(:,k),'linewidth',1.5) %Trq_sim_sfunct
plot(simOut2(k).Trq.Time,t_sim_n(:,k),'linewidth',1.5) %Trq_sim_original
```

```
plot(file(k).time_line,t_typ(:,k),'lineWidth',1.5) %w_ref_typ
legend('T_{sim}_{ref}','T_{sim}','T_{typ}','location','northeastoutside')
title('Torque')
xlabel('s')
ylabel('Nm')
cd figure
cd Plot_double_sim
    cd(filenames(k).name)
    saveas (figure(4*(k-1)+2),['Torque.svg'])
    saveas (figure(4*(k-1)+2),['Torque.mfig'])
cd ..
cd ..
cd ..

figure(4*(k-1)+3)
hold on
grid on
plot(simOut(k).Vdq_ref.Time,Vd_sim(:,k),'linewidth',1.5) %Vd_sim
plot(simOut(k).Vdq_ref.Time,Vq_sim(:,k),'linewidth',1.5) %Vq_sim
plot(file(k).time_line,Vd_typ(:,k),'lineWidth',1.5) %Vd_typ
plot(file(k).time_line,Vq_typ(:,k),'lineWidth',1.5) %Vq_typ
legend('Vd_{sim}','Vq_{sim}','Vd_{typ}','Vq_{typ}','location','northeastoutside')
title('Vdq_{ref}')
xlabel('s')
ylabel('V')
cd figure
cd Plot_double_sim
    cd(filenames(k).name)
    saveas (figure(4*(k-1)+3),['Vdq.svg'])
    saveas (figure(4*(k-1)+3),['Vdq.mfig'])
cd ..
cd ..
cd ..
```

```
figure(4*(k-1)+4)
hold on
grid on
plot(simOut(k).ldq_fb.Time,Id_sim_fb(:,k),'linewidth',1.5) %Id_sim_fb_sfun
plot(simOut(k).ldq_fb.Time,Iq_sim_fb(:,k),'linewidth',1.5) %Iq_sim_fb_sfun
plot(file(k).time_line,Id_fb_typ(:,k),'lineWidth',1.5) %Id_typ_fb
plot(file(k).time_line,Iq_fb_typ(:,k),'lineWidth',1.5) %Iq_typ_fb
legend('Id_{sim}_{sfunct}','Iq_{sim}_{sfunct}','Id_{typ}','Iq_{typ}','location','northeastoutside')
title('Idq_{fb}')
xlabel('s')
ylabel('A')
cd figure
cd Plot_double_sim
    cd(filenames(k).name)
    saveas (figure(4*(k-1)+4),['Idq_fb.svg'])
    saveas (figure(4*(k-1)+4),['Idq_fb.mfig'])
cd ..
end
```

The following table reports all the Simulation made, with in the different parameter required. Thanks to the automatize script all the data have been collected and stored with just one click. It's important to notice that the S-Function's script thanks to the already C-Code available is 10 times faster than the original script, and this simply reduce the time of simulation form 10 hours to just 6 (5 occupied form the original script only). This because it's been considered 10 minutes for each simulation and plotting that is reduce to 1 min in case of S-Function availability, so 2^5 simulations for each case means 64 simulations overall that means more than 10 hours.

			Simulation Parametres			Drivers requires		Model Parametres		
Simulation ID	Typology of Simulation	Environment	Ts [s]	Tsc [s]	duration [s]	Torque demanded [% of T_max]	Speed required [rpm]	Dumping Factor [Nms]	max slope [rad/s^2]	I max charge/dis. [A]

1	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	100%	6000	0,076	500	300
2	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	100%	6000	0,076	500	500
3	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	100%	6000	0,076	900	300
4	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	100%	6000	0,076	900	500
5	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	100%	6000	0,114	500	300
6	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	100%	6000	0,114	500	500
7	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	100%	6000	0,114	900	300
8	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	100%	6000	0,114	900	500
9	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	100%	9000	0,076	500	300
10	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	100%	9000	0,076	500	500
11	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	100%	9000	0,076	900	300
12	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	100%	9000	0,076	900	500

13	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	100%	9000	0,114	500	300
14	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	100%	9000	0,114	500	500
15	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	100%	9000	0,114	900	300
16	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	100%	9000	0,114	900	500
17	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	75%	6000	0,076	500	300
18	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	75%	6000	0,076	500	500
19	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	75%	6000	0,076	900	300
20	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	75%	6000	0,076	900	500
21	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	75%	6000	0,114	500	300
22	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	75%	6000	0,114	500	500
23	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	75%	6000	0,114	900	300
24	Virtual Time Mode	Typhoon	2E-06	4E-05	1,5	75%	6000	0,114	900	500

25	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	75%	9000	0,076	500	300
26	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	75%	9000	0,076	500	500
27	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	75%	9000	0,076	900	300
28	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	75%	9000	0,076	900	500
29	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	75%	9000	0,114	500	300
30	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	75%	9000	0,114	500	500
31	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	75%	9000	0,114	900	300
32	Virtual Time Mode	Typhoon	2E-06	4E-05	2,5	75%	9000	0,114	900	500

4.3. Real Time Simulations: Typhoon and Closed Loop building

In this second part the Real-Time Simulation is approached. Once again, one of the simulations is reported (in this case #3) but, in this case only Typhoon simulations have been used. So, the MATLAB script, just explained has been exploited to plot only the two cases of the same Simulation in Typhoon, both, the Virtual and the Real Time Mode with the same conditions.

These RTM simulations can't be conducted with the same clock-time used for the Plant (Ts = $2e^{-6}$ s) this is since the CPU of the host PC can't go this fast. So, simply the SCADA PANEL underlines the "Overflow error" and doesn't complete the simulation properly. The Ts used is changed to $10e^{-6}$ s

(the minimum value to avoid the Overflow is around $8e^{-6}$ s). Moreover, to avoid any possible trouble with the Nyquist theorem or "non-multiplicity" between the Ts and Tsc, the Tsc value has been increased also from the initial value of $40e^{-6}$ s to $100e^{-6}$ s. In this way 10 times higher than Ts.

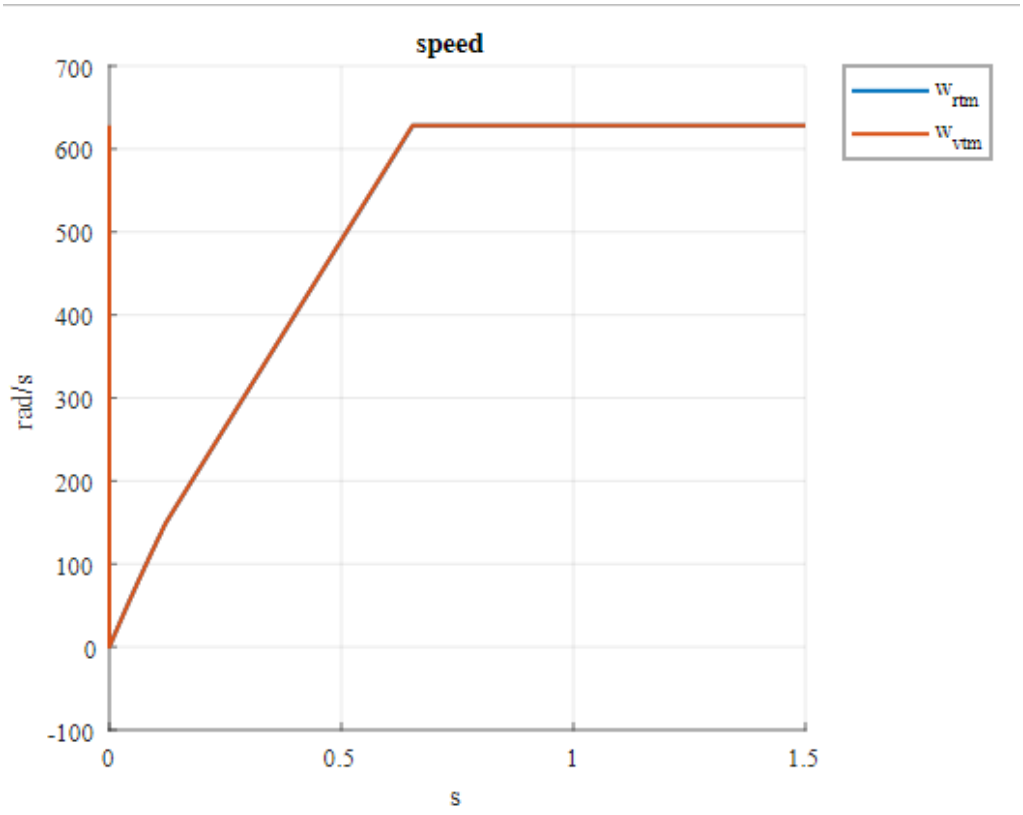

Figure 4.10 Speed

Apart from the changed conditions simulated there is no notification to do in this graph.

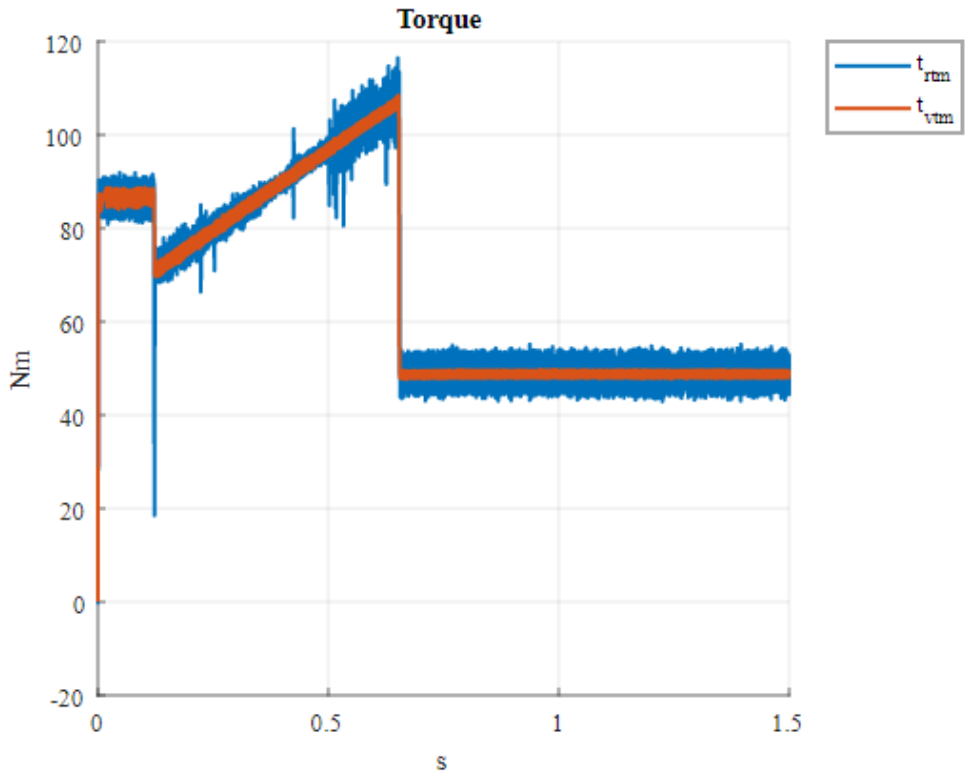

Figure 4.11 Torque

About Figure 4.11 it's possible to see that the RTM runs correctly but with a higher Tsc, indeed the noise visible is much higher than in the Virtual case. Also, during the speed reference controlling phase there are many unstable peaks that are the consequence of non-correct Tsc.

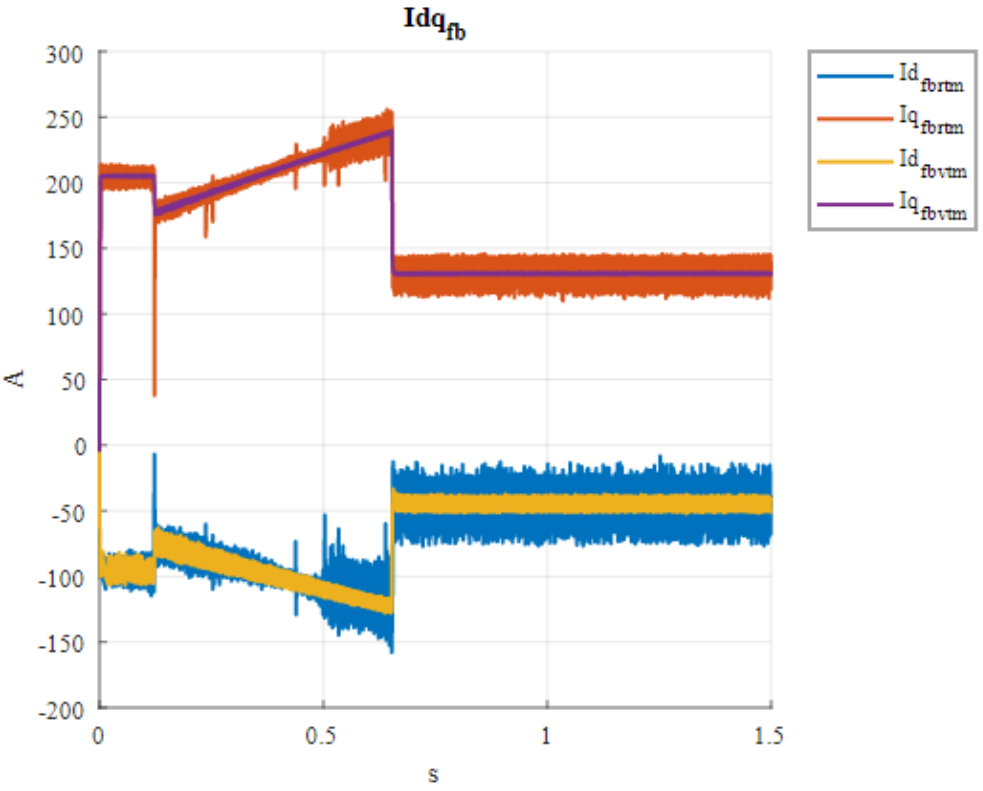

Figure 4.12 feedback currents

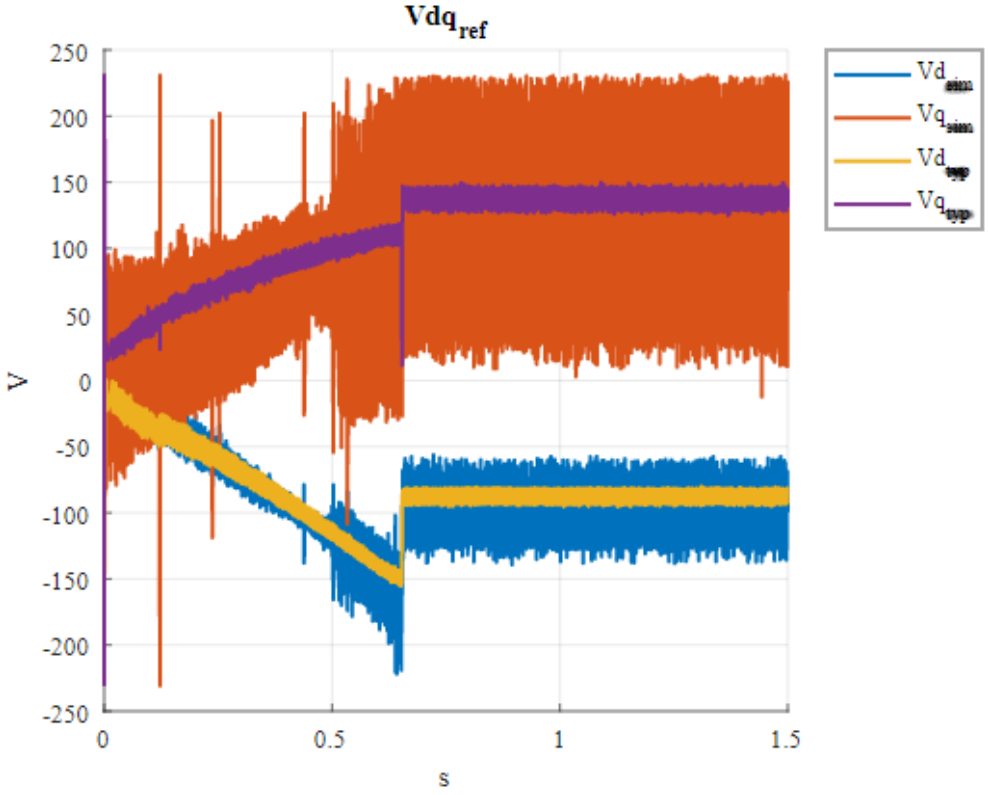

Figure 4.13 Voltages: feedback voltages (The Legend is wrong, see Figure 4.12)

In this second the graphs analog considerations can be done, included the same considerations done in all Virtual cases.

In the three following paragraphs two more developed have been apported to the system and this has been necessary to overcome the issue reported here about the CPU limitation. Moreover, the next steps also, improve the system itself introducing the real concept of Hardware in the Loop development. Indeed, Typhoon has been developed to be embedded with so many different third-part devices included many different integrated Evaluation Board. To better going into these commercial details is suggested to have a look at the Typhoon HIL website[42] but also the free documentation produced[43]. The example in the video tutorial suggests using the Texas Instruments Evaluation Board LAUNCHXL-28379D. By the way, the approach used for this work has been little different. Indeed, as will be explained in the last section, a SPARK ECU (Engine Control Unit) from Alma Automotive has been used to simulate the mathematical algorithm of the controller. This approach makes the development much harder because a third-part software (LabVIEW) is needed to pilot the Board.

4.3.1. HIL validation example for Closed-Loop Simulation

As first step an "artificial" HIL loop has been built: In this section only a very simple example is introduced just to make understand the differences that occur between this phase and the previous one.

A square signal has been generated, to make the example be observable, it's been chosen to use a very high frequency for the signal: 1 MHz. The generated signal is than passed directly to the scope "output signal" available in the SCADA Panel. But, before been plotted is amplified by a direct gain that simply double its amplitude. Also, the signal is in parallel connected to the Analog output pin #1 and the same signal is connected in loop with the analog input #15 on the same board. The example has been tested in the 3 following cases: the scan time imposed is the smallest available for this device configuration ($5e^{-7}$ s).

[42] https://www.typhoon-hil.com/methodologies/

[43] https://www.youtube.com/watch?v=-FxyBRd8-YA&t=203s&ab_channel=TyphoonHIL

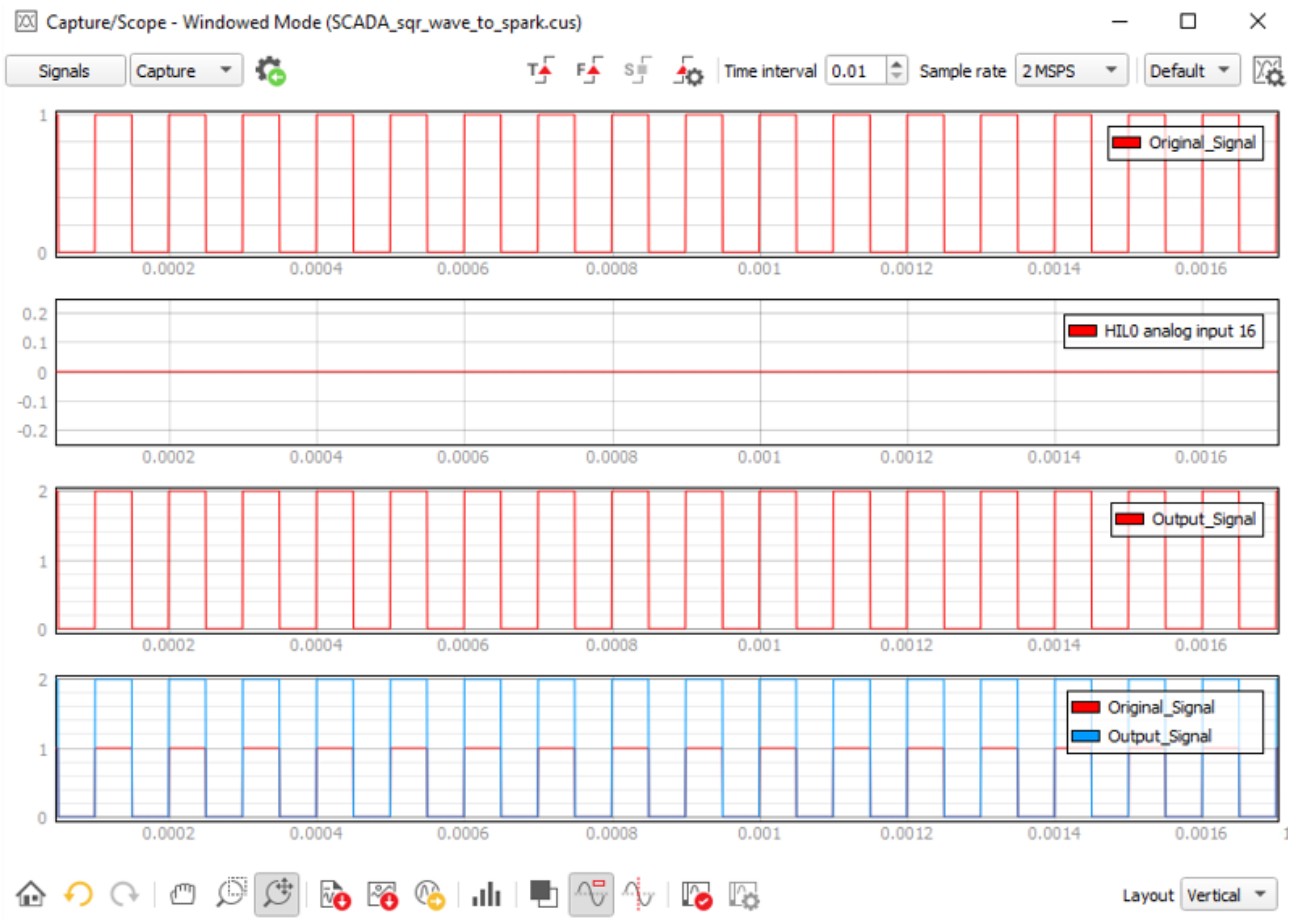

Figure 4.14 Only Virtual Case.

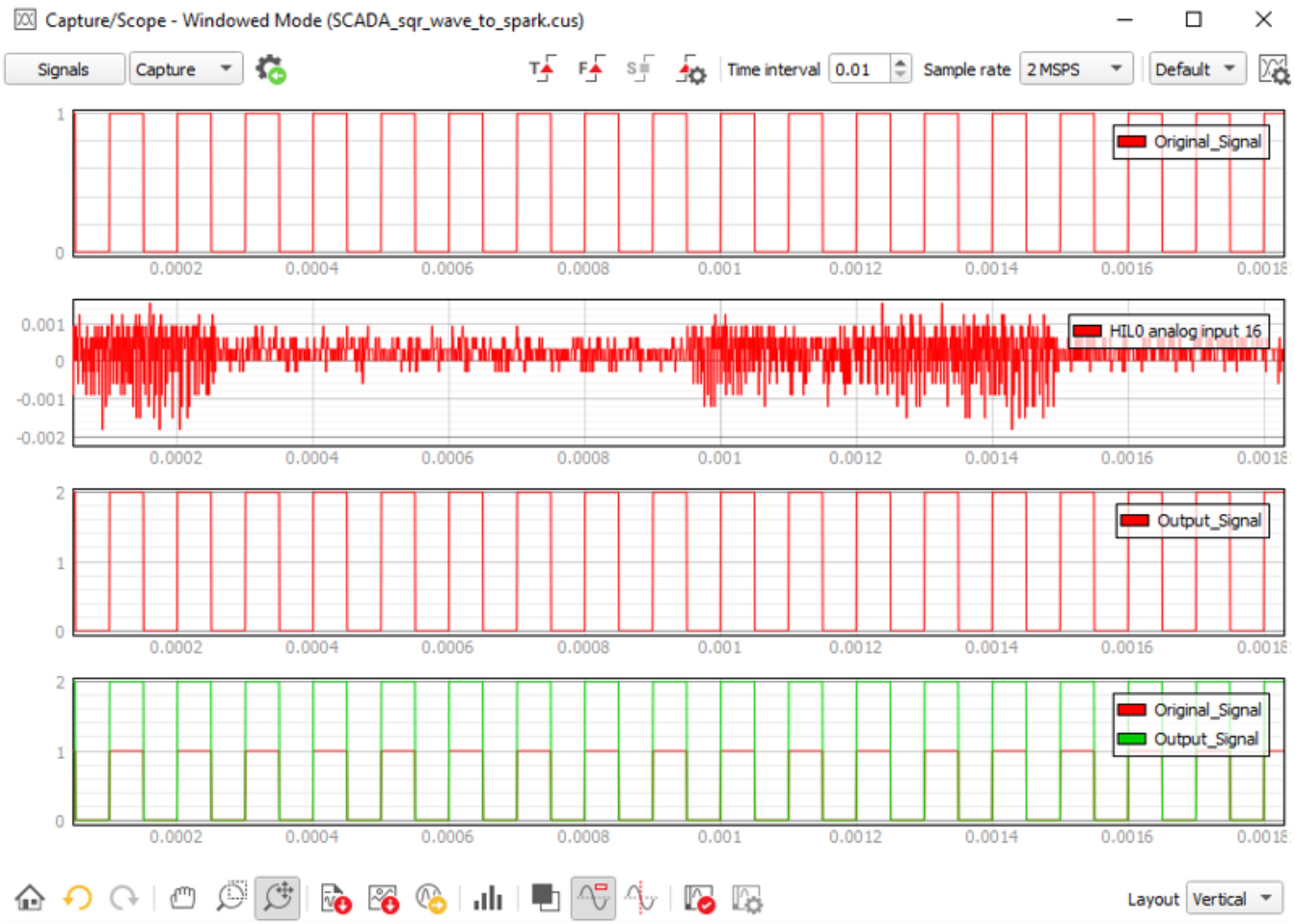

Figure 4.15 Real-Time Simulation without connecting physically the I/O pins.

Figure 4.16 Full-Real-Time Simulation, with physical signals.

The three picture above represents the three different situations simulated on the same script.

In

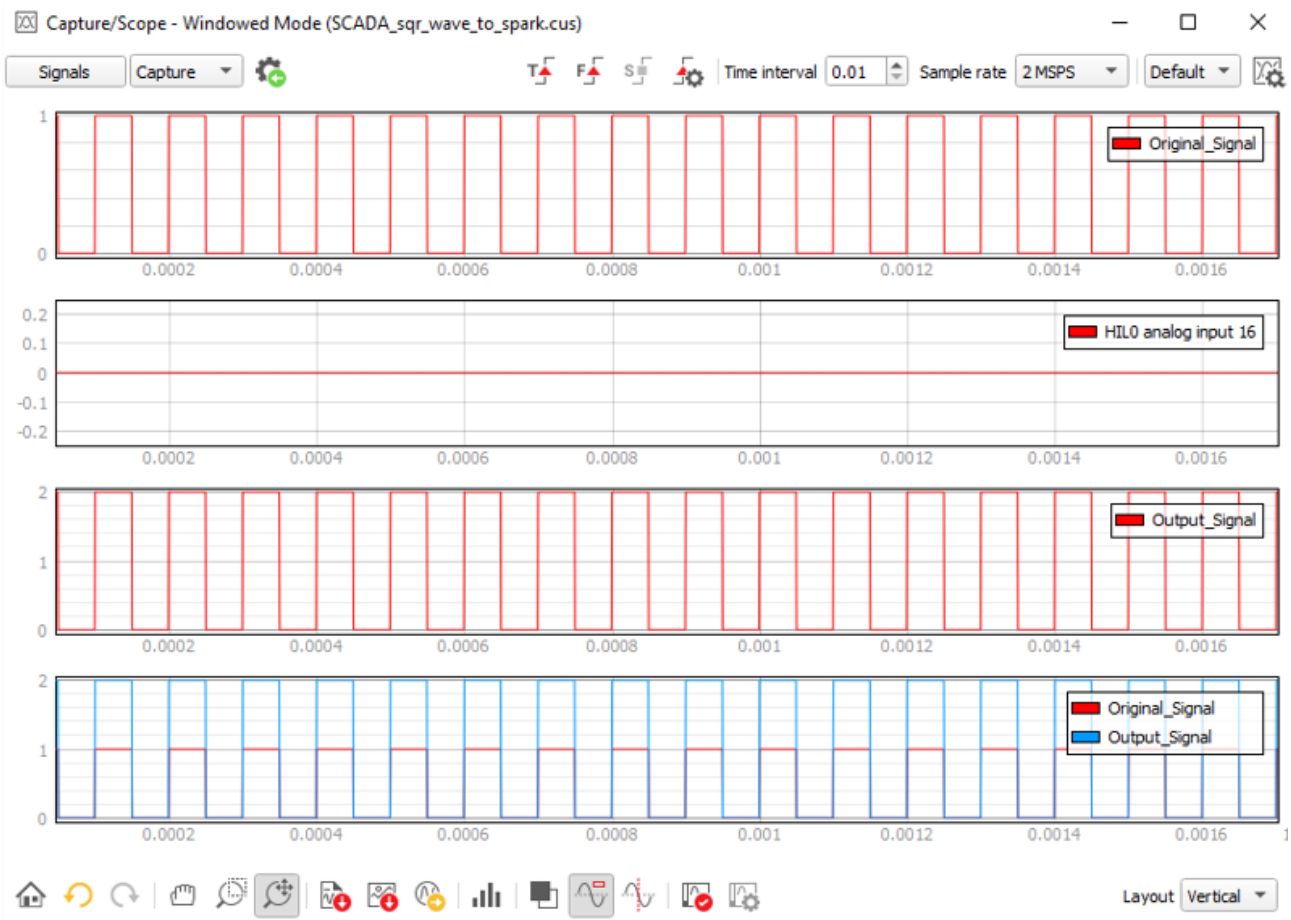

Figure 4.14, a total Virtual simulation is reported. What is happen is exactly what can be the

expectations of a theorical analysis. Being no physical signal form the pins, no HIL 0 input is reveled.

In

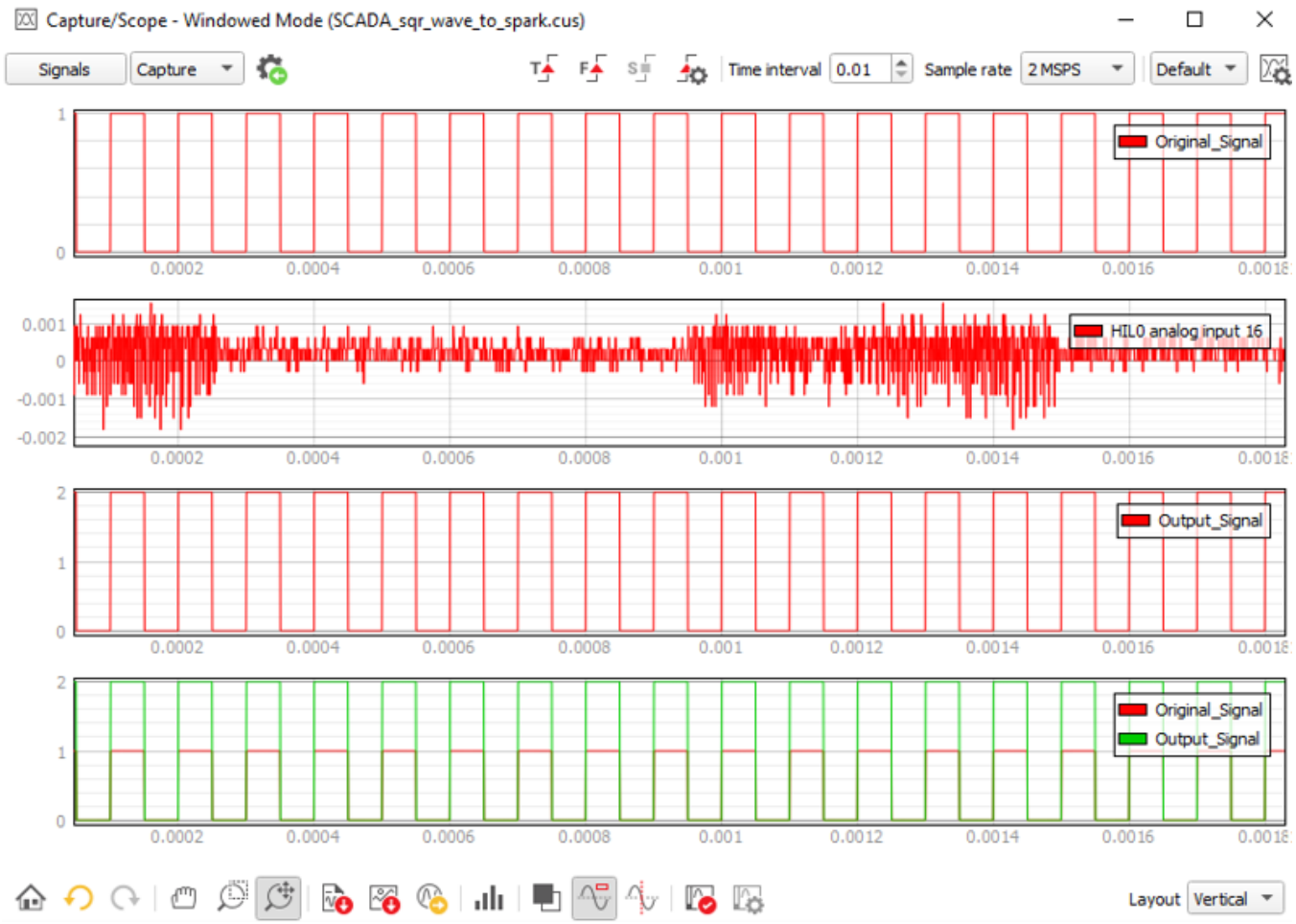

Figure **4.15**, a Real-Time simulation is conduced. So, the breakout board is now turned on, but in any case, no signal is connected to the physical pin. This justifies what is shown: only some

disturbance read by the board, probably because the simulation has been conducted in a noise room. All the other graphs are the same shown in the Virtual case.

In the last

Figure 4.16, Real-Time simulation with physical signal is observed: now it is clear that introducing the physical signal a certain delay must be taken into consideration. In this very case, the simplicity of the general architecture tested makes the delay to be visible only with a very fast simulation (1 MHz) and, also in this situation the delay is barely visible in terms of $10e^{-6}$ s, that is a very short reference frame.

4.3.2. Preliminary Closed-Loop with limitations

In these two last paragraphs the last achieved results is taken into consideration: the simulation trough the HIL test bench is preliminary done using only software Typhoon HIL and its embedded device HIL 602+. Later, the integration of the usage of the Linux Real-Time OS is discussed.

• About the Real-Time conditions: the Hardware in the Loop is firstly tested only the embedded board of the HIL 602+. In this case, the virtual simulation is modified in this way: the feedbacks signals are substituted with **analog input** signals and the outcomes of the controller are passed to their relative **analog outputs.** These signals are physically connected "in loop" in the embedded board. So, the real inputs are the 4 feedbacks, and the 3-reference voltage signals the drives the inverter and vice-versa the same 7 output pins are connected in loop.

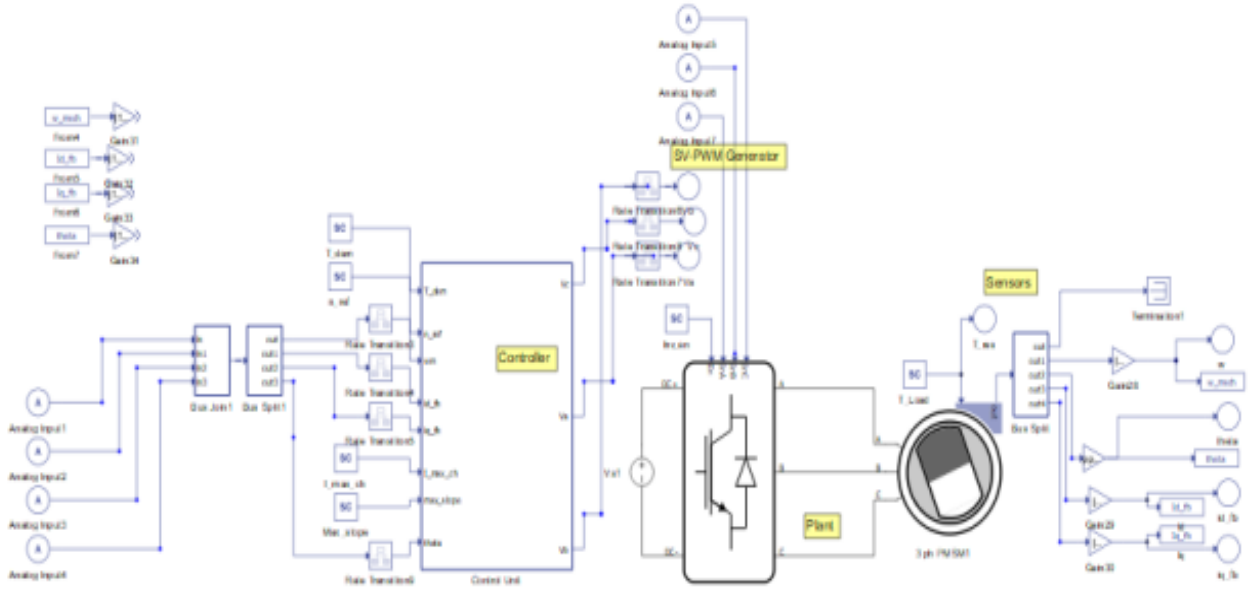

Figure 4.17 shows the modified Typhoon model, where it is possible to notice, in evidence, the 7 I/O pins exploited.

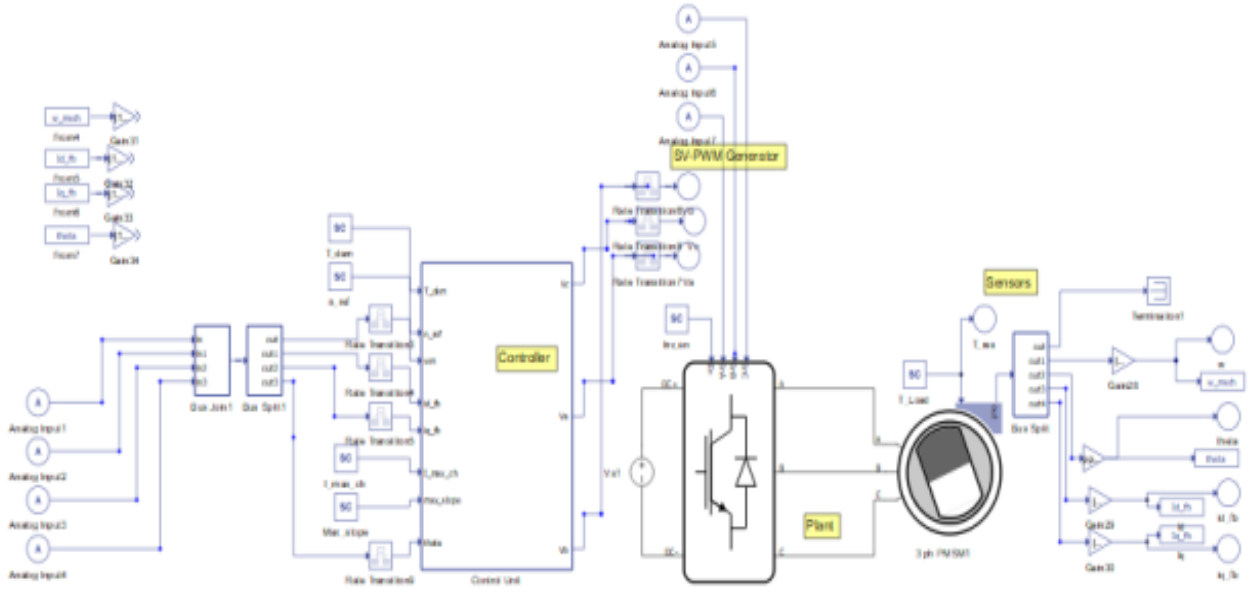

Figure 4.17 Complete Model: analog inputs (feedbacks + user references) and outputs (3-pahse command signal for the inverter).

Of course, the output pins cannot be present in the Schematic Editor, so they must be configured how showed in the following screenshot from the SCADA Panel (

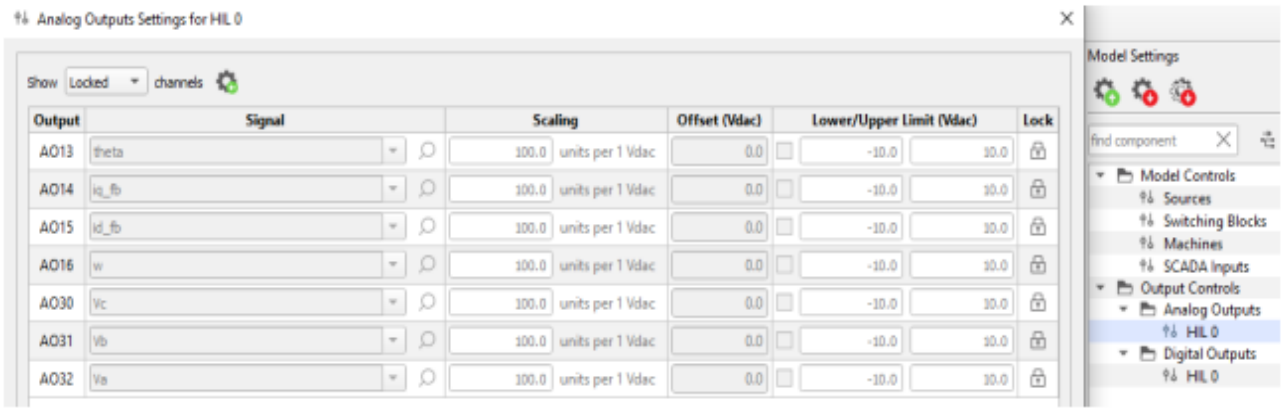

Figure 4.18) are visible the number of pins involved, and the scaling for the representation used.

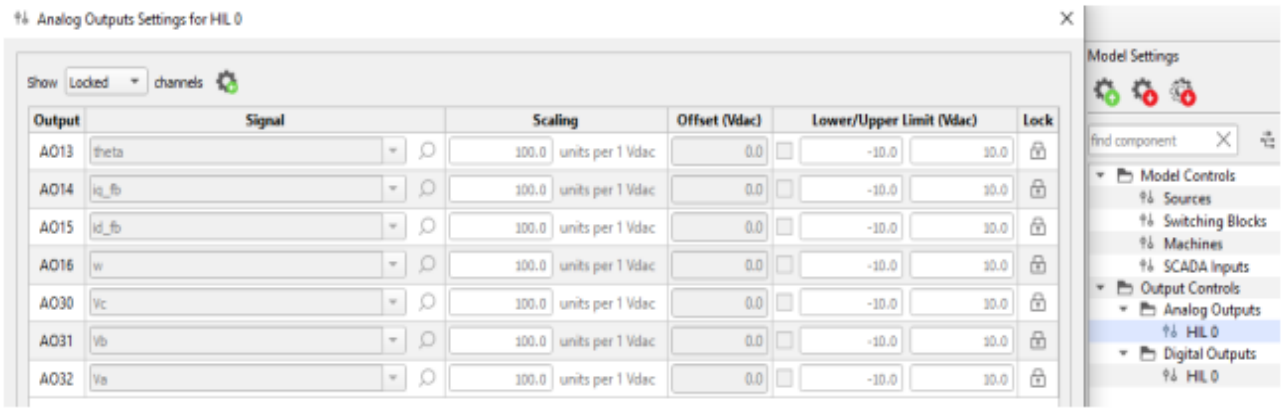

Figure 4.18 Analog Outputs configuration interface.

In the end, the complete simulation with this modified architecture is run. In

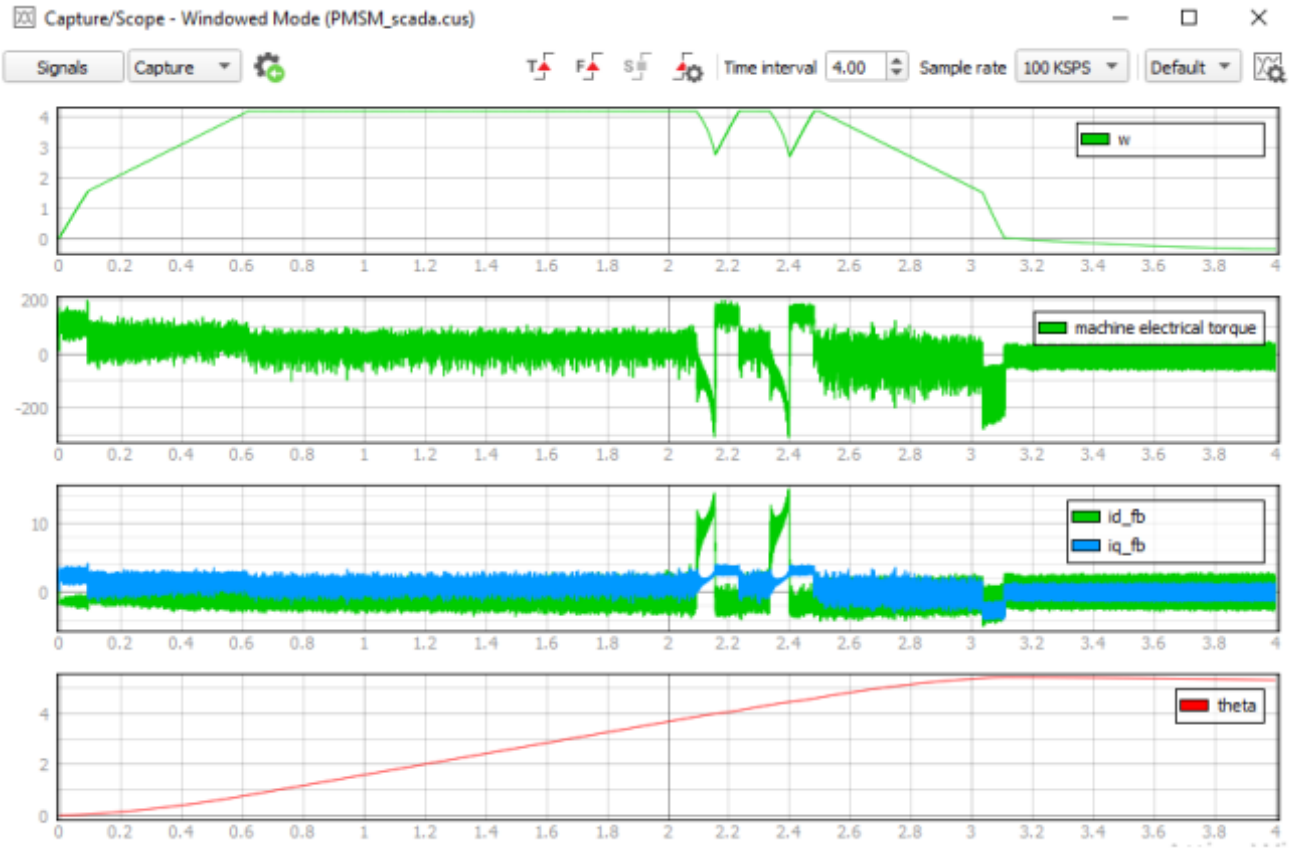

Figure 4.19 the feedback signals are reported, this time, only the Capture-Scope block has been used, this is necessary to reduce the CPU utilization. Indeed, the main problem of this architecture is not the non-complete HIL built but the limitations imposed by the CPU performances.

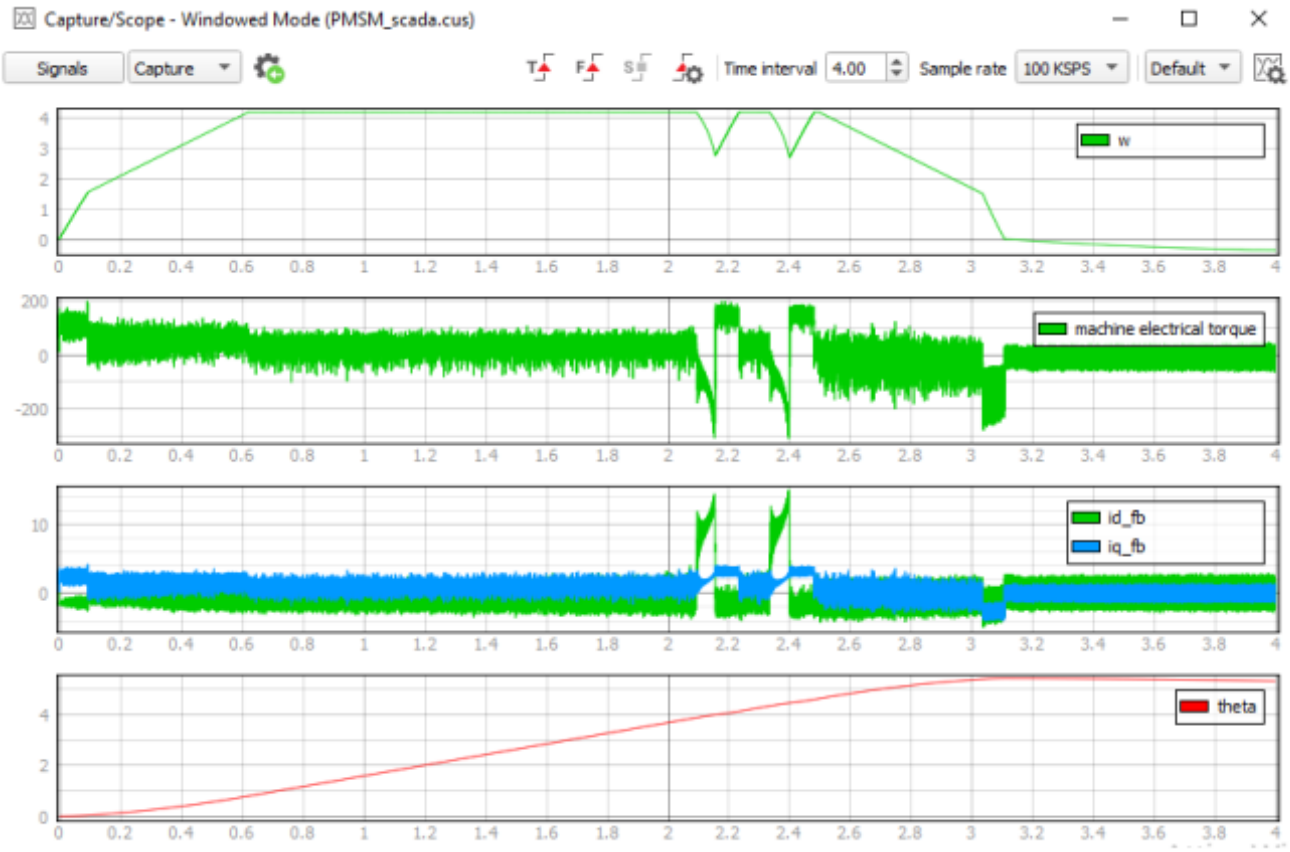

Figure 4.19 Complete HIL Simulation: angular speed, electrical torque, currents, and angle.

- From

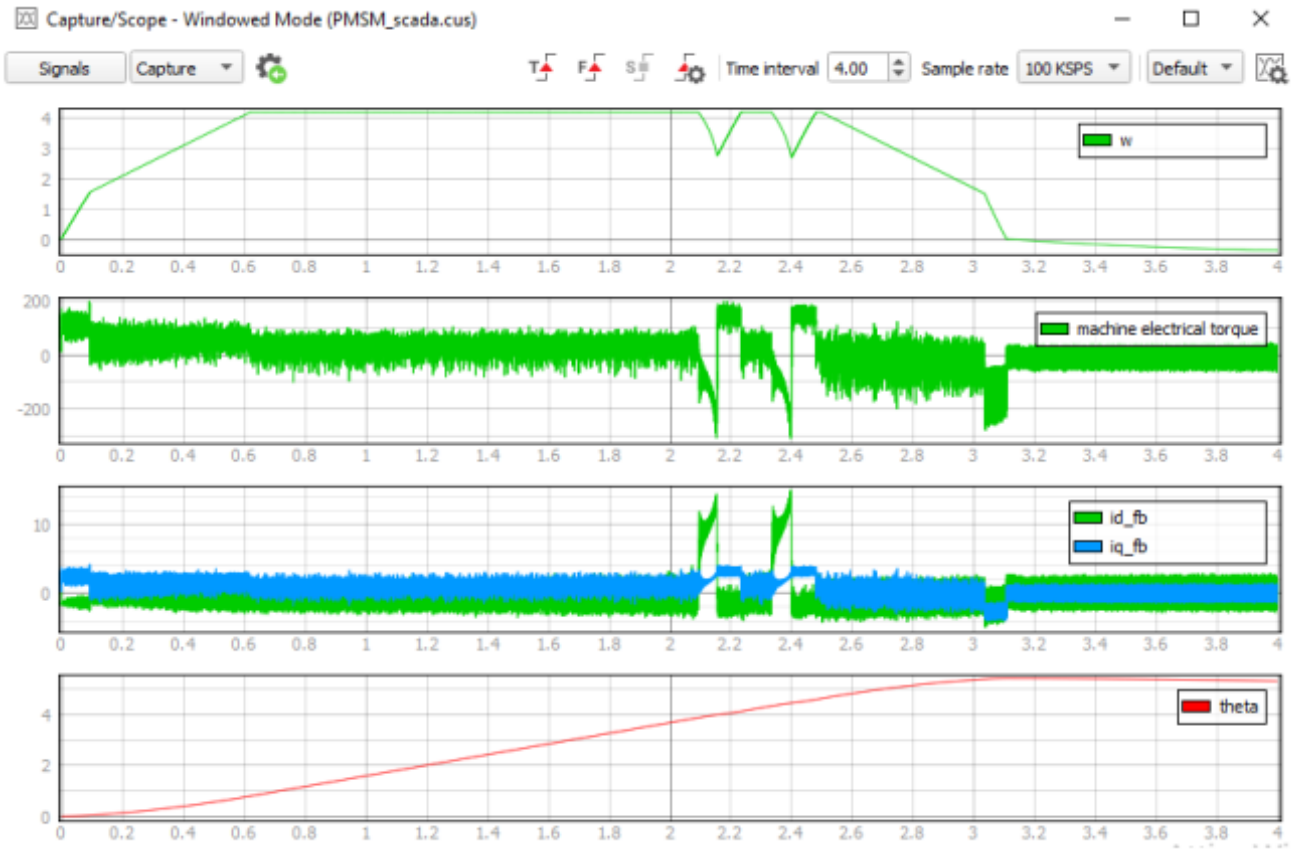

Figure 4.19 is possible to notice that the simulation result is not correctly conduced, in two main different spots the speed reference imposed (that for this time has been reduced to 4000 rpm) cannot be followed in RT. When this is verified, strange peaks are visible in the torque and currents graphics;

- This condition, changes every-time the simulation is run, this, verify that the CPU maximum performance has been reached;
- Also, the scan-times parameters have been changed from the nominal ones, this is necessary to have at least this “un-precise outcomes”: Ts = 10 e^{-6} s and Tsc = 60 e^{-6} s;
- In this simulation, also the scale visible in the ordinates are not correct, indeed, as previously said the scaling factor on the signal output is imposed to 100 units/V. So, to read the correct valued read in its nominal scale it’s, once again necessary to impose a gain of 100 units/V. This, of course can be easily done directly in the plotting phase, so no useless calculation are imposed to the CPU, already fully exploited.

4.3.3. Closing the Loop with SPARK ECU strategy

Now, about what has been defined as “Closed Loop HIL architecture” also the SPARK ECU board has been involved. The Linux Real-Time OS is now used to substitute the host PC CPU to overcome the performance limitation discussed. To do this, a more complex architecture is needed:

- Trough the software LabVIEW a special .so library is imported with the “Call a Library Function Node” to which the parameters of the function “Step” in the C code are passed. This Node appears exactly as the same way discussed in the S-Function building for Simulink case (see the dedicated paragraph).

- To pass the I/O analog signals to the controller a special FPGA VI is needed as interface. In the main VI two special nodes called “refers to FPGA VI” are used to open the reference for a FPGA VI that basically works as a function. These nodes used are two different branches because two different FPGA sub-VI are needed: the first one for open the “read VI” that is used to get the inputs from the pins on the board; the second one is for “write VI” that is used to pass the analog outputs to the pins on the board. Inside the relative FPGA VIs two different VIs are created and deployed, indeed the FPGA VI must be compiled before being “called” from the main VI inside the RT Module. See the blinking red Led inside the 2.4.1. paragraph to better clarify how the architecture works.

- The two FPGA interfaces above introduced are easily building because they just need to use the blocks “***read***”, “***write***” and “***enable***” that are referred to each pin. The main difficult however is the impossibility to know the Component Level IP (CLIP) file that contains the information to map the pins present on the breakout board. These files can be created with the relative wizard available in LabVIEW, or alternatively already created socket CLIPs can be used. In this case, the company ALMA also provided the CLIP socket with the relative FPGA VIs already usable, however all the files are password-protected, this makes impossible to understand how they works. So, only the “read FPGA VI” has been used for two main reasons: since the front panel is visible the names of the pins used can be read; but also the breakout board has only 2 pins for analog outputs and this create an issue in the very case of this work because the 3-phase reference voltages needs 3 pins to be passed to the Typhoon breakout board. This last issue can be easily overcome using one of the mathematical transformations CLARK or PARK already discussed, this because, how it is well known, a wave needs only to data to be represented: a magnitude and a frequency.

- With reference to the previous point, in this last one the read FPGA VI is tested. Directly on the FPGA target the named “ADC_read” VI shared from ALMA is run to read a slow wave generated

with an external signal generator (Figure 4.20 b). Also, in Figure 4.20 a) the same signal is read with an oscilloscope. What can be notice form this test are two main things:

- The impossibility to have a graphical representation of the wave signal because the password protected VI can't be edited, so no graph can be added;
- From some empirical reads of the signal read by the FPGA VI contains a gain of -0.5 that basically helve the magnitude and inverse the sign.

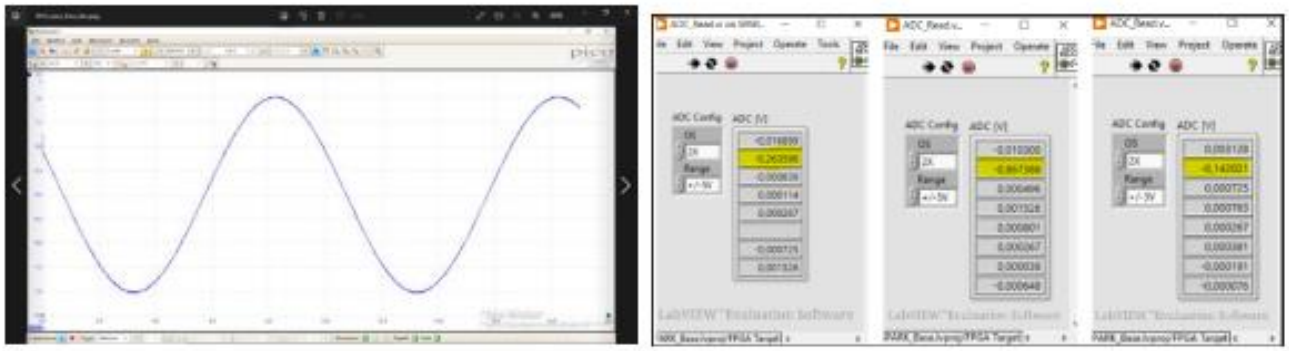

Figure 4.20 a) Wave signal read with the Oscilloscope: (Magnitude 3.4 V; frequency 0.1Hz). b) Wave signal read from the ADC_read VI on the second (in yellow) read from differential pins on the breakout board.

In the following Figure 4.21 three more screenshots are reported. In these pictures, this time the signal comes from the Typhoon HIL 602+ breakout board. The slow square signal is plotted respectively: the generated signal in the typhoon SCADA Panel, the signal read from the oscilloscope and the "ADC_read" FPGA VI with the breakout board of the SPARK ECU board (this time, in yellow, the first differential channel is used). These figures are used as a validation of the correct communications between the two boards. Once again, this time it is easier to notice, that the -0.5 gain is content in the VI.

Figure 4.21 a) Signal generated in Typhoon: (Magnitude 1 V; 1 Hz frequency). b) Signal read with the oscilloscope. c) Signal read with the VI on LabVIEW.

In this last part, in

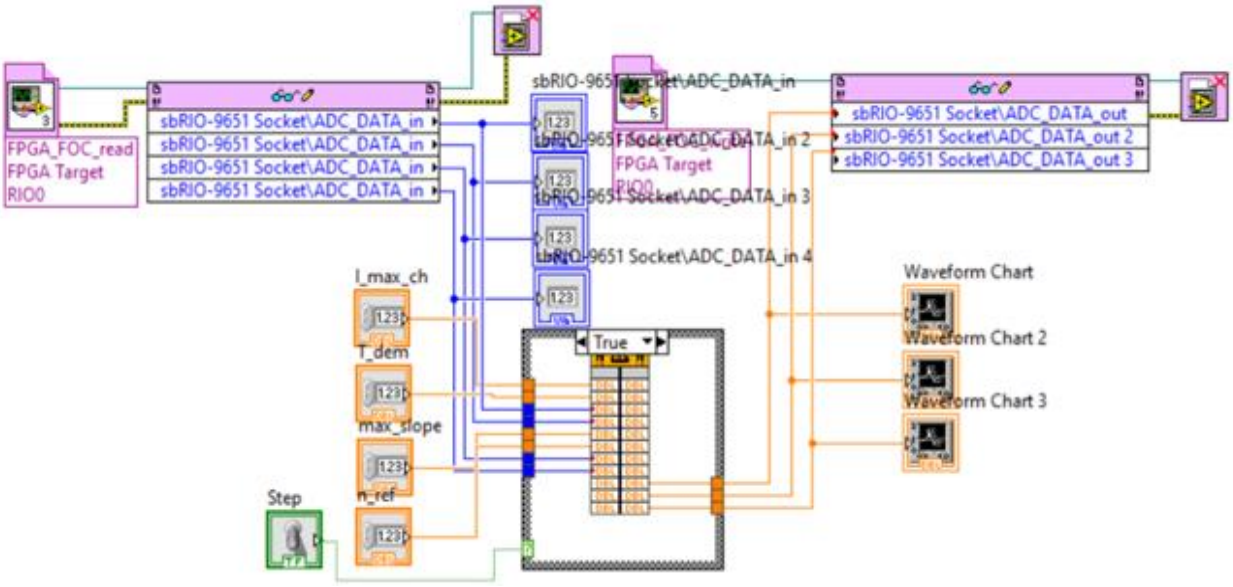

Figure 4.22 is reported the entire model from the Diagram Block on LabVIEW where:

- On the left, the read-FPGA reference is open, and the analog signal can be read;
- On the right, the write-FPGA reference is open, and the analog signal are passed to the outputs pins on the board;

- Both reference nodes are linked to the central node (that is a “case” structure) where the respectively outputs are connected to the closing reference node are called;
- Inside the case structure, in both the T/F configuration the same “Call a Library Function Node” is inserted, but respectively the ***step*** and the ***init*** wrapper function are settled. The inputs are basically the parameters that must be passed to the function (so, 4 feedbacks read as AI and 4 parameters settled by the user); the outputs are the 3 3-phase voltage signal that must be passed to the inverter on the Plant (Typhoon breakout board).

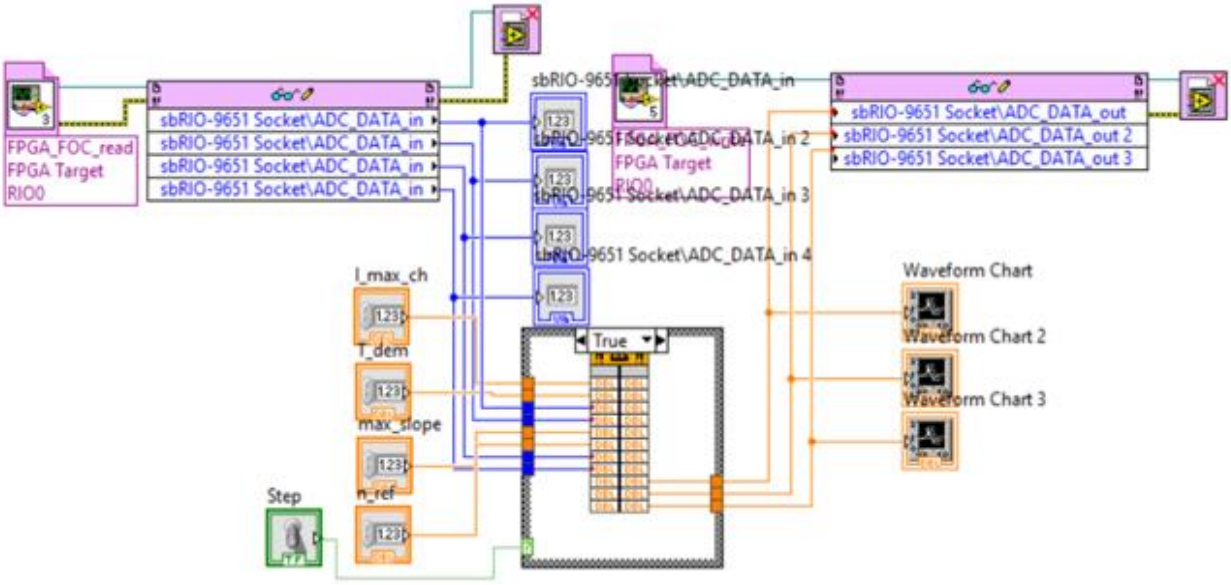

Figure 4.22 Block Diagram for the entire FOC controller.

In Figure 4.23 is also shown the Front Panel where it is possible to find:

- 4 user-defined signals (Simulation parameters: see the table on paragraph 3.1);
- 4 feedback signals in input to the controller;
- The digital toogle user-defined to switch the case structure between the two configurations to change the function called (step or init) from the “call a library function node”;
- On the right, the 3 graphs represents the plots of the 3-phase voltage output signals.

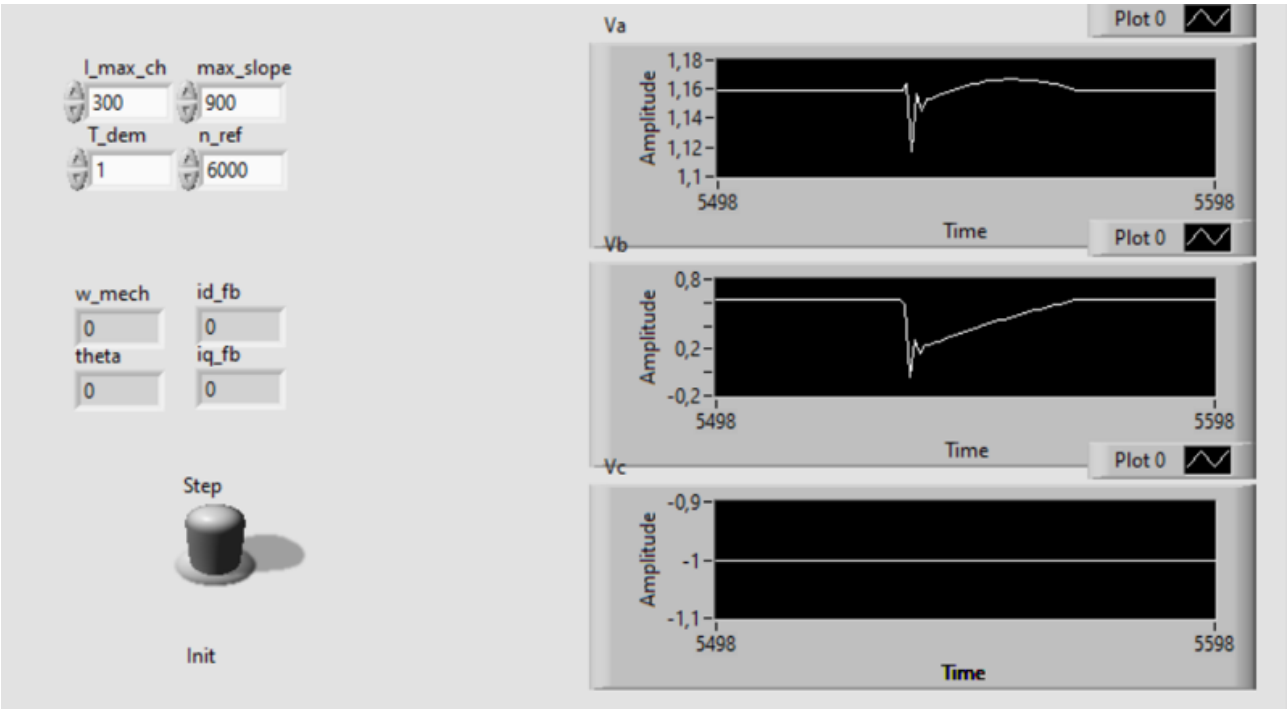

Figure 4.23 Front Panel of the entire FOC controller.

In

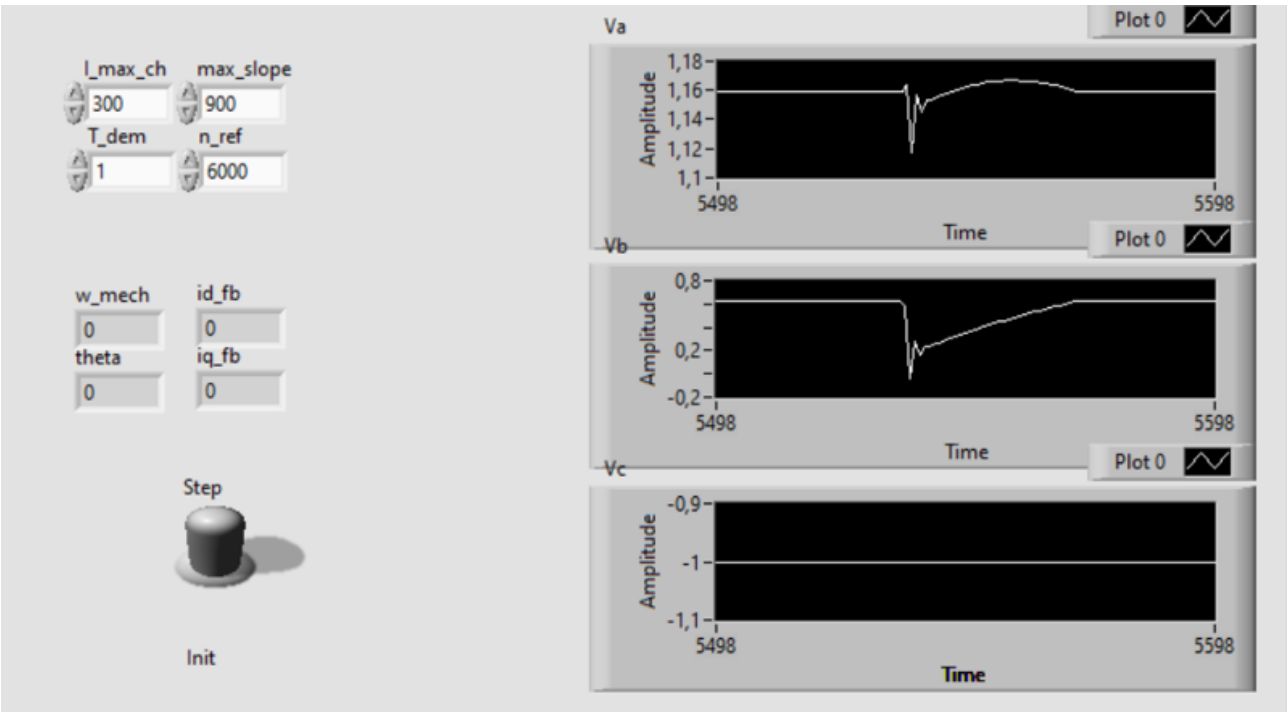

Figure 4.23, has been run a simulation that wants to test the correct behavior of the node “call a library function”. At the beginning of the simulation the toogle is set to “init” so all the parameters are set to zero, till the toogle is untouched nothing happens. When the step function is activated simply the step function start to run but and this is visible in the graphics. However, the outputs is immediately

reported on the zeros because in this configuration no physical signals are connected to the board, so simply the pins read only zeros.

So, exactly has said for the S-Function case, the difference between the two function is:

- Init: Initialize the variables to zero (or in the case of fixed parameters to their nominal values);
- Step: runs “effectively” the Field Oriented Control logic of control contained inside the C Code.

Lastly, the following Figure 4.24 photo report a summary of the all architecture directly involved in the test of the bench inside the Teoresi’s offices.

Figure 4.24 The overall architecture of the test bench on the desk of the Teoresi’s offices.

From left to right: The Host PC connected to the device Typhoon HIL 602+, the breakout board has in output the analog signals (plant’s feedbacks), The very same signals are the inputs on the differential pins of the breakout board of NI connected to the SPARL ECU device, The board is also connected (via-ethernet) to the second host PC where the LabVIEW software is run to use the exploit

the Linux RT OS where the control logic is computed. In the picture also the signal generator, the oscilloscope and the voltage source for the NI board are visible.

5. Conclusions

As introduced in the abstract, the aim of this work is to exploit the Hardware in the Loop technique to build an embedded test bench. Starting from the available literature about the Field Oriented technique applied to a Permanent Magnets Synchronous Motor which can be considered as the "case of study", many different goals have been achieved:

-Building the architecture in the MATLAB/Simulink development environment and testing its affordability, dealing with Flux-Weakening conditions, a battery management system and tuning the control parameters.

- Exploit many features available in the software Typhoon HIL and its embedded device HIL 602+ to import the algorithm, test it in both Virtual and Real Time Mode. Also, exporting the data of the simulation conducted, but, most importantly the C-Code which computes the controller algorithm.

- Manipulate the C-Code to test the same control algorithm with different system architectures. In particular developing an S-Function for MATLAB, a library exploited from a Linux Real-Time system and, more in general, to create a work-flow to make this validation process generalizable and easily replicable.

-Through the testing of the main "case of study" do an analysis about the computational performances reachable from each of the architectures developed. Also bring to the attention, for each architecture the limitations and the difficulty introduced from each of the SW or HW involved.

Lastly, it can be said that the main goal of this work is the closure of an Hardware in the Loop architecture trough the complete removal of all of the performance constraints due to the available SW and HW. So, re-introducing the economical aspect, as previously discussed in the very beginning the aim of this work is to try to give information and proofs about the better choices available when it's time to approach a problem of development or validation of a control code when it's needed to use an Hardware in the Loop test bench, that is nowadays a very common approach for those works that needs to reduce cost and resources waste.

5.1. Limits of the model and possible improvements

About the very algorithm, some other considerations can be done about the possibility to improve the weakness analyzed in the previous chapters. It can be convenient to explicit these different themes one by one:

1- As mentioned, the Source of the power has been simplified a lot, indeed only an ideal generator as been used. It's obvious that create a more sophisticated model could be a good improvement in the affordability of the bench in terms of reality. This can be done, including many different phenomena that make the Battery model more realistic. For example: thermal phenomena, realistic limitation in term of power supplied, usage limitation and more can be introduced. Of course, these approximations make also the BMS a very simplified model of what can be a "realistic" simulation of a true battery.

2- Also, the theoretical approach has its limitations, indeed when, in Chapter 1 has been discussed the magnetic flux behavior, to simplify the equations, their linearity has been imposed as a "acceptable approximation" in the case discussed despite the current values taken into consideration were not completely in the linear field of the graph shown. In this case, entire thesis or papers tries to give a realistic characterization of the magnet filed behavior, this themes, are far from my path of study, but of course, more sophisticated models could be used. Moreover, empirical data of the characteristic can be also introduced, substituting the simplified equations, for example using a Look-Up-Table approach to utilize non-linear physics equation.

3- Speaking about Look-Up-Tables, can be helpful for a better approximation but also to reduce the peaks shown by the simulations to re-build a more accurate LUT, but considering also to use more than one parameters in the building process: indeed also the switching speed could be a parameter in the computation and not just a fixed value (let's think about the driving style), but also, introducing a model also in the disturbance parameter, till now imposed as a constant. In this way basically the 1D-LUT experimented could be seen a 3D-LUT.

4- The stability of the PID parameters used have been proved to been "solid", by the way introduce any algorithms/techniques or tool to have a better chance of tuning the parameters cannot be a bad idea.

Index of Figures

Bibliography

- Amin, F., et al. (2017, May), Modelling and Simulation of Field Oriented Control based Permanent Magnet Synchronous Motor Drive System, In Indonesian Journal of Electrical Engineering and Computer Science, 6(2).
- Chin, Y., & Soulard, J. (2003), A Permanent Magnet Synchronous Motor for Traction Applications of Electric Vehicles, Presented at IEEE Electric Machines and Drives Conference, Madison, Wisconsin, United States.
- Cypress Semiconductor (2017), Retrieved from Coordinate Transform in Motor Control.
- Ferraris, L. (1999), Macchine Elettriche, Turin, Italy: CLUT.
- Ferreira, F., Ge, B., Quispe, E., & De Almeida, A. (2014), Star- and Delta-Connected Windings Tolerance to Voltage Unbalance in Induction Motors, Presented at International Conference on Electrical Machines (ICEM), Berlin, Germany.
- Fitzgerald, A., Kingsley, C., & Umans, S. (2003), Electric Machinery (Sixth Edition), New York, New York, United States: McGraw-Hill.
- E.Giammello (2018/2019) - Simulazione e test HiL del controllo di azionamenti elettrici in ambienti MATLAB e PLECS – Politecnico di Torino
- Khan W.A. (2016), Torque Maximizing and Flux Weakening Control of Synchronous Machines, Espoo, Finland School of Electrical Engineering.
- Krishnan, R. (2010), Permanent Magnet Synchronous and Brushless DC Motor.
- Drives, Boca Raton, Florida, United States: CRC Press.
- Mohan, N., Undeland, T., & Robbins, W. (2003), Elettronica di Potenza. Convertitori e Applicazioni (Terza Edizione), Milano, Italy: Hoepli.
- Ohm, D. (2000), Drivetech Inc., Retrieved from Dynamic Model of a PM Synchronous Motor.
- Vezzaro M.(2009/2010) Caratterizzazione sperimentale di un motore sincrono a magneti permanenti mediante la misura dei flussi in presenza si saturazione incrociata, Università degli studi di Padova, Italy.
- Rossi, R (2018-2019). Design of a PMSM Field-Oriented Control Algorithm.
- L.Salvatore – Motore sincrono a magneti permanenti – Politecnico di Bari
- Toshiba Electronic Devices & Storage Corporation (2018), Retrieved from DC-AC Inverter Circuit.
- (Mathworks.com)
- (NI.com)
- (Typhoon-hil.com)
- (Wikipedia.org)