Politecnico di Torino

Master's Degree in Mechatronic Engeeniring

A.a. 2022/2023

Modelling of an electric induction motor on FPGA for testing in HIL environment

Relatori:

Prof. Alessandro Rizzo

Ing. Antonio Vitale

Candidati:

Luca Mutton

Abstract

The automotive world is in a new evolution phase, and the need for clean mobility lead in the last decade to an electrification process for vehicles. This new impulse in researching and developing hybrid electric vehicles (HEV) and battery electric vehicles (BEV) requires increasingly accurate tools to evaluate electric powertrain performance and reliability. For this purpose, the simulation of real physical components in a virtual environment offers an easier way to perform tests and analyses on a real component, saving cost and development times.

In this context, the Hardware in the loop (HIL) can perform real-time simulation in a virtual environment fully programmable. This offers a new testing technique for electronic control units, simulating partially or entirely the system. As an example, an engine control unit (ECU) can be tested without connecting it to a real inverter or even to an electrical machine; that's thanks to the FPGA boards that can reach higher frequencies than a classic processor; frequencies that are needed when simulating inverters or motors at high rpm. Another important feature of the HIL is the possibility to combinate real components with the simulated ones, to avoid modelling cheap components like relays and sensors.

This thesis aims to create a parametric model of an asynchronous Induction Motor and a Three-phase Inverter and validate them. The fact that the models are parametric is quite important because a simple change of parameters can obtain a different behaviour of the model corresponding, for example, to a new motor. Hence the advantages of this testing method are evident. The simulation Hardware used to achieve the thesis objective is the Scalexio real-time processor from dSPACE and for the implementation, software such as MATLAB, Vivado and dSPACE RCP/HIL, all provided by KINETON S.r.l., a company specialized in simulation and validation in the automotive sector.

Politecnico di Torino

List of content

Links

1. https://www.electronics-tutorials.ws/rc/rc-integrator.html
2. https://roboticsknowledgebase.com/wiki/system-design-development/In-Loop-Testing/
3. https://www.dspace.com/en/pub/home/products/hw/simulator_hardware/scalexio.cfm#175_54760
4. https://en.wikipedia.org/wiki/Field-programmable_gate_array
5. https://en.wikipedia.org/wiki/AC_motor
6. https://www.xilinx.com/htmldocs/xilinx14_7/sysgen_gs.pdf
7. https://en.wikipedia.org/wiki/Direct-quadrature-zero_transformation
8. https://en.wikipedia.org/wiki/Alpha%E2%80%93beta_transformation
9. https://it.mathworks.com/help/mcb/gs/implement-motor-speed-control-by-using-field-oriented-control-foc.html
10. https://www.elprocus.com/difference-between-open-loop-closed-loop-control-system/
11. https://it.mathworks.com/solutions/power-electronics-control/field-oriented-control.html?s_eid=PSM_15028
12. https://it.mathworks.com/videos/reinforcement-learning-for-field-oriented-control-of-a-permanent-magnet-synchronous-motor-1587727861081.html
13. https://it.mathworks.com/solutions/power-electronics-control/clarke-and-park-transforms.html#:~:text=Le%20trasformate%20di%20Clarke%20e%20Park%20vengono%20comunemente%20utilizzate%20per,frame%20stazionario%20ortogonale%20(%CE%B1%CE%B2)
14. https://www.tntech.edu/engineering/pdf/cesr/ojo/asuri/Chapter3.pdf
15. https://www.tntech.edu/engineering/pdf/cesr/ojo/asuri/Chapter4.pdf
16. https://inverterdrive.com/file/WEG-132kW-315Frame-2Pole-B5

Figures

1. https://www.evgo.com/ev-drivers/types-of-evs/#bevs
2. https://it.mathworks.com/help/ecoder/gs/v-model-for-system-development.html
3. https://www.dspace.com/en/pub/home/products/hw/simulator_hardware/scalexio.cfm#175_54760
4. https://www.latticesemi.com/en/What-is-an-FPGA
5. https://www.automate.org/blogs/what-is-a-brushless-dc-motor-and-how-does-it-work

6. https://thors.com/wp-content/uploads/2022/03/electric-motor-fundamentals-course.jpg
7. https://en.engineering-solutions.ru/motorcontrol/pmsm/
8. https://commons.wikimedia.org/wiki/File:Couple_glissement_MAs.svg
9. https://silvaco.com/blog/design-ip-for-automotive-socs-trends-and-solutions/
10. https://it.mathworks.com/help/mcb/gs/open-loop-and-closed-loop-control.html
11. https://microchipdeveloper.com/asp0107:settling-time-overshoot
12. HTTPS://WWW.RESEARCHGATE.NET/FIGURE/SOFTWARE-IN-THE-LOOP-SIL-AND-HARDWARE-IN-THE-LOOP-HIL-AUTHORS-L-NYKA-AND-J_FIG3_341453328
13. https://club.auto-doc.it/magazin/che-cose-linsonorizzazione-unauto
14. https://www.epaddock.it/come-fatto-e-come-funziona-il-motore-della-motoe/
15. https://it.frwiki.wiki/wiki/Machine_asynchrone
16. https://auto.hwupgrade.it/news/trasporti-elettrici/bosch-presenta-il-nuovo-motore-elettrico-per-furgoni-e-camion-medi-800-volt-e-inverter-col-99-di-efficienza_101088.html
17. https://microchipdeveloper.com/asp0107:settling-time-overshoot
18. https://gomechanic.in/blog/ecu-electronic-control-unit-explained/
19. https://it.mathworks.com/help/mcb/gs/open-loop-and-closed-loop-control.html

List of figures

Introduction

Thesis Objective

This project aims to demonstrate the potential and the advantage of the testing method in a virtual environment called Hardware in the Loop (HIL). The goal consists of implementing a Three-phase RC Inverter and a Squirrel cage Induction Motor, to test a real external Engine Control Unit (ECU) that works with a sensored Field Oriented Control (FOC). In absence of a real controller, for simulation purposes, we also modelled a very simple Controller.

Due to the high frequencies that could be encountered in this type of application, some of the models will be configured for an FPGA board. The Models are modelled to be full parametric, to make them feasible for numerous types of motors and inverters. Furthermore, the application, which in our case is for automotive power trains, thanks to the previous feature is very flexible resulting in a simple change of parameters.

In this work, we will explain step-by-step the workflow followed by me and my collaborator to achieve our goal: starting from the study and explanation of the theory behind the models, analysing also the critical issues encountered during the building process, to the final working models and their results.

The project's practical activities were performed in collaboration with Kineton S.R.L. a consultant company in the automotive sector that gives us the tools (hardware and software) and the knowledge we need for the competition of this work.

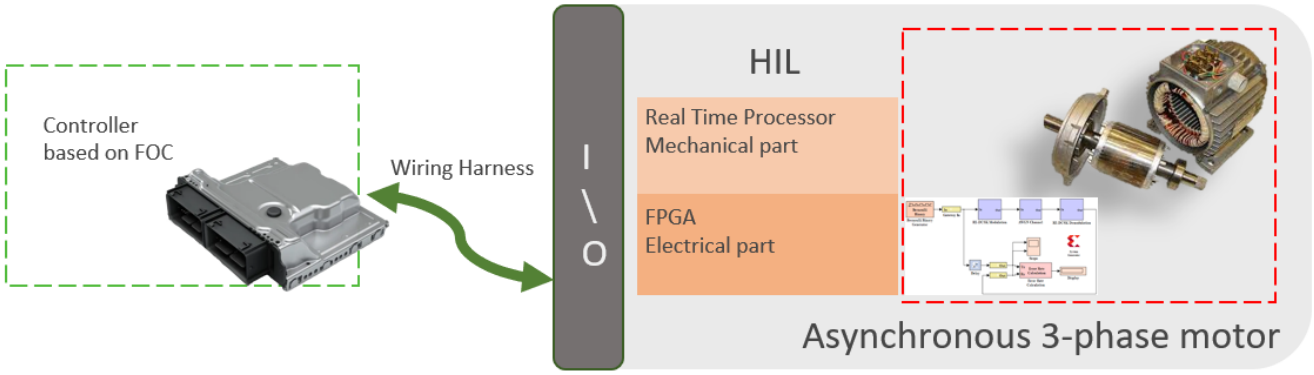


Fig. 1 General model for the HIL

Automotive

The increasing awareness towards the impact of pollution on the environment has made it necessary in every sector to move away from traditional fuels. So, the automotive world is in a new evolution phase, and the need for clean mobility lead in the last decade to an electrification process for vehicles. Connectivity, infotainment, traction, and security, every aspect of a vehicle had electrical change, from the past, as shown in the figure below.

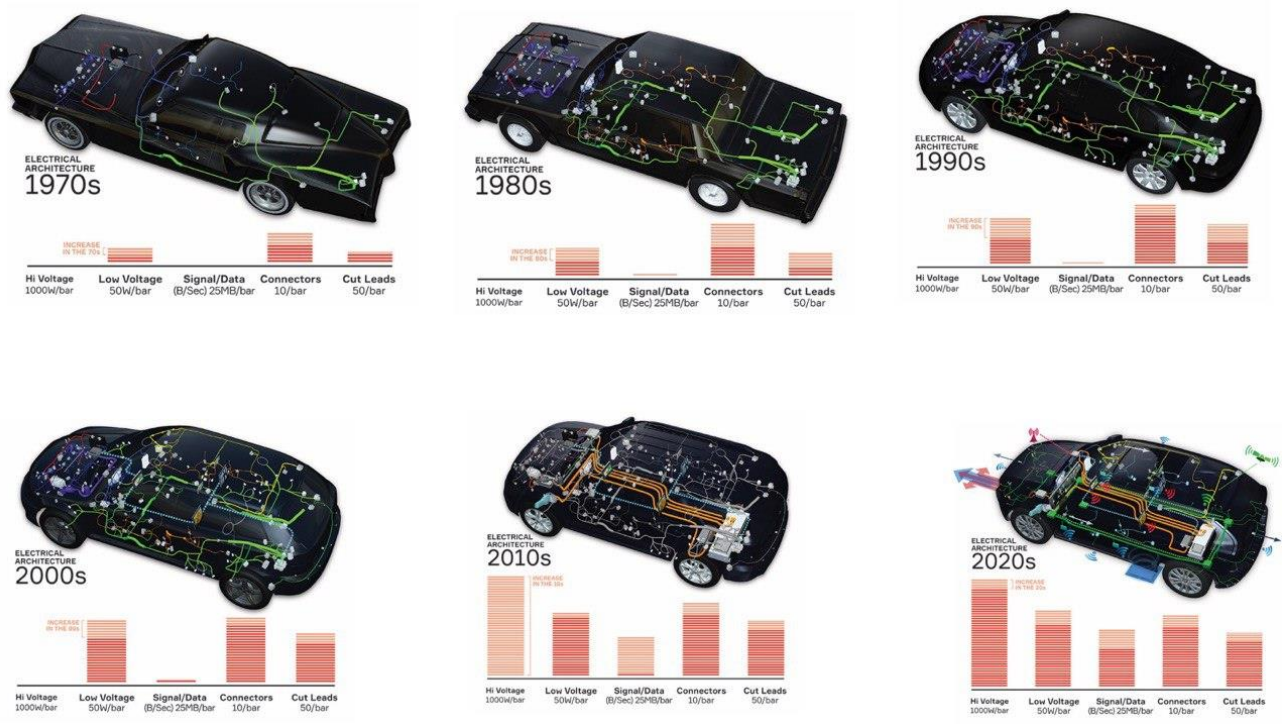


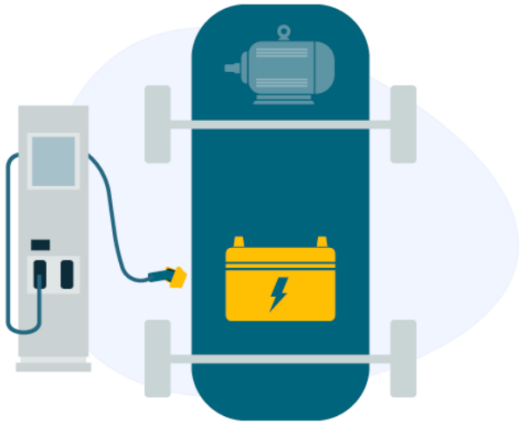
Fig. 2 electrical parts in a vehicle

Electric vehicles (EVs) represent now the future evolution of the automotive world. EVs use electricity or a mix of gasoline and electricity as a power source, aiming to reduce the total emission of the vehicles.

Nowadays in the market, there are three main categories of electric vehicles (EV):

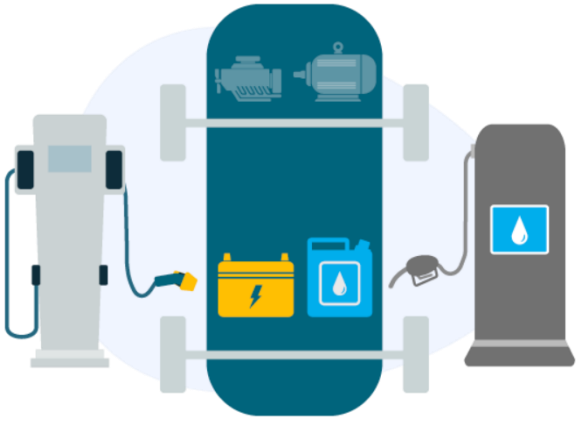
- **Battery Electric Vehicles (BEVs)**

  Entirely powered by an electric battery and an electric motor for traction.



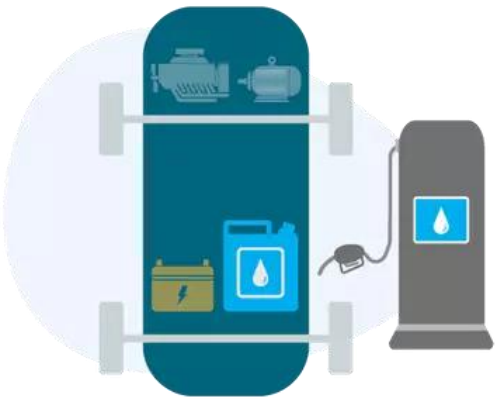
- **Plug-in Hybrid Electric Vehicles (PHEVs)**

  Has two power sources, a battery and a fuel tank, and both can be charged and refuelled. Both motors can be used for traction.



- **Hybrid Electric Vehicles (HEVs)**

  In this case, all the power comes from gasoline. The electric motor is used only to assist the internal combustion engine.



Verification and validation

As complexity is reaching higher and higher levels in every component of the vehicles, the need for testing the control algorithm and control device has become a key process in developing vehicles. Verifying and validating system development is fundamental, and the V-model represent an approach widely spread in every engineering field.

The V-Model is a system development representation that highlights the verification and validation steps in the system development process. The left side of the "V" identifies the steps leading to code generation, including system specifications and detailed software design. The right side of the V focuses on verifying and validating the steps mentioned on the left side, including software and system integration.

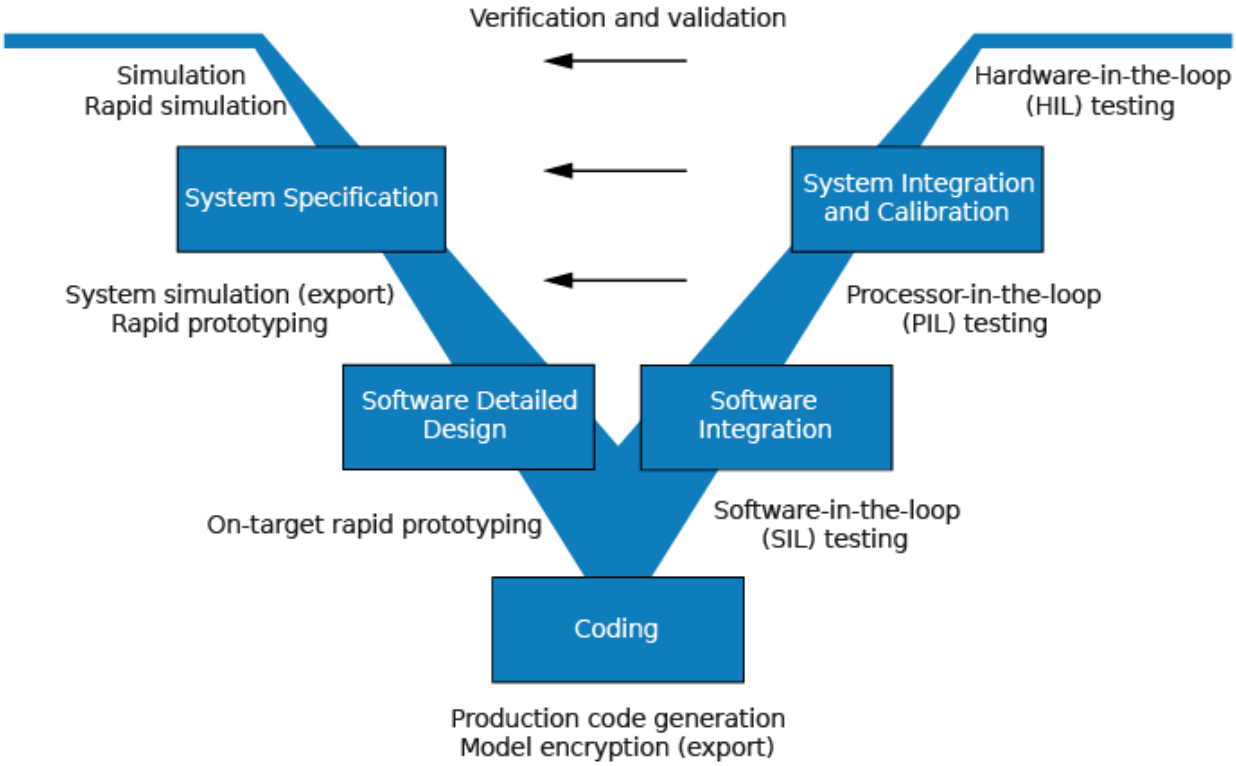


Fig. 3 V-Models

There are three main methods for testing and validating a model:

- **Software in the loop (SIL)**
  This is a method for testing and validating code in a simulated environment in order to detect bugs quickly and inexpensively and improve code quality.

- **Processor in the loop (PIL)**
  This refers to the testing and validation of the embedded software on the processor which will later be used in the electronic control unit (ECU).

- **Hardware in the loop (HIL)**
  This refers to the techniques of verification (testing) of electronic control units, consisting of connecting them to special benches that reproduce in a complete way the electrical and electronic system of the system for which they are intended.

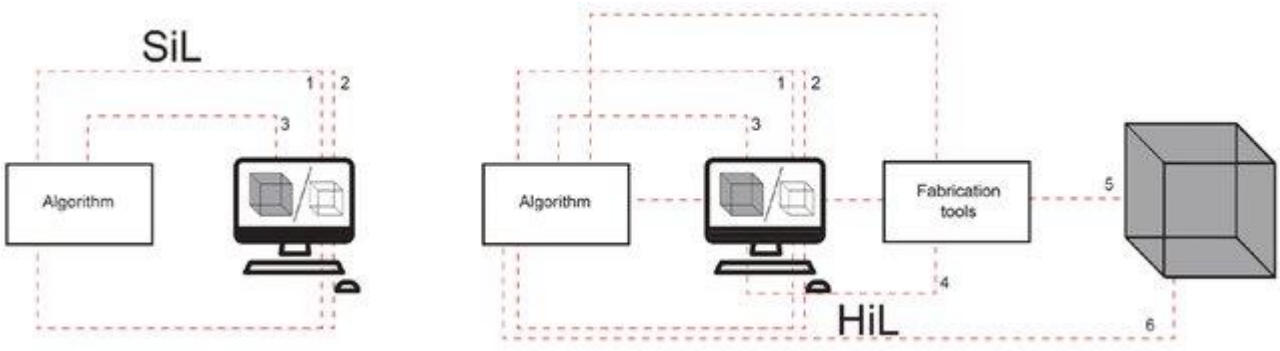


Fig. 4 SIL vs HIL

Hardware in the loop

Hardware in the loop (HIL) is a testing methodology that can be used to test specific types of real-time operating components, such as ECUs, reducing costs and time for a test and improving its effectiveness. This new testing technique was made necessary by the increasing complexity of the hardware under test, as the number of tests to be performed keeps increasing.

This testing method can partially o entirely replaces the system, and the complexity of the plant needed to make the test effective and comparable to the real test is achieved by adding a mathematical representation of all related dynamic systems. In this way, a simulation can replace real complex systems, such as vehicles, so that even in the early development stage It can be tested embedded components without having the entire system.

Advantages

The advantages of the hardware in the loop testing method are numerous:

- **Time:** developing the machines and the control systems in parallel can cause errors to be found in during the commission, with HIL errors can be found earlier and resolved earlier, to reduce commission time.

- **Costs:** Real plants to be tested could be expensive, and often there is more than one prototype before the final one. This increase start-up cost and require expensive safety precautions. HIL fits perfectly for resolving this problem, even though the initial design can take many hours, the implementation requires less time, and the parametrised models can be reused for different applications.

- **Safety: t**esting beyond the normal range of operations or in failure conditions, are fundamental for revealing whether the control system can operate the machine safely, these tests can be dangerous. HIL simulations take place in a virtual environment, which makes safety almost negligible.

- **Quality:** finding problems and errors earlier during the development process instead arranging some solutions at the final stages of the design, increasing the overall quality.

- **Human factor:** it can be used to test human interaction with machines, this allows you to test whether the machines can be used easily and comfortably. It can also be used to train operators.

Hardware

The hardware used for this project comes from the dSPACE company, which produces high-performance simulators.

Fig. 5 dSPACE SCALEXIO Family

SCALEXIO is a flexible and modular systems specifically designed in hardware-in-the-loop (HIL) and rapid control prototyping (RCP) applications for various industries such as automotive, automation, aerospace, medical, transportation, or research. It can perform Real-time simulations that suit for the most demanding applications and offer extreme flexible connectivity with the numerous I/O module that can be installed.

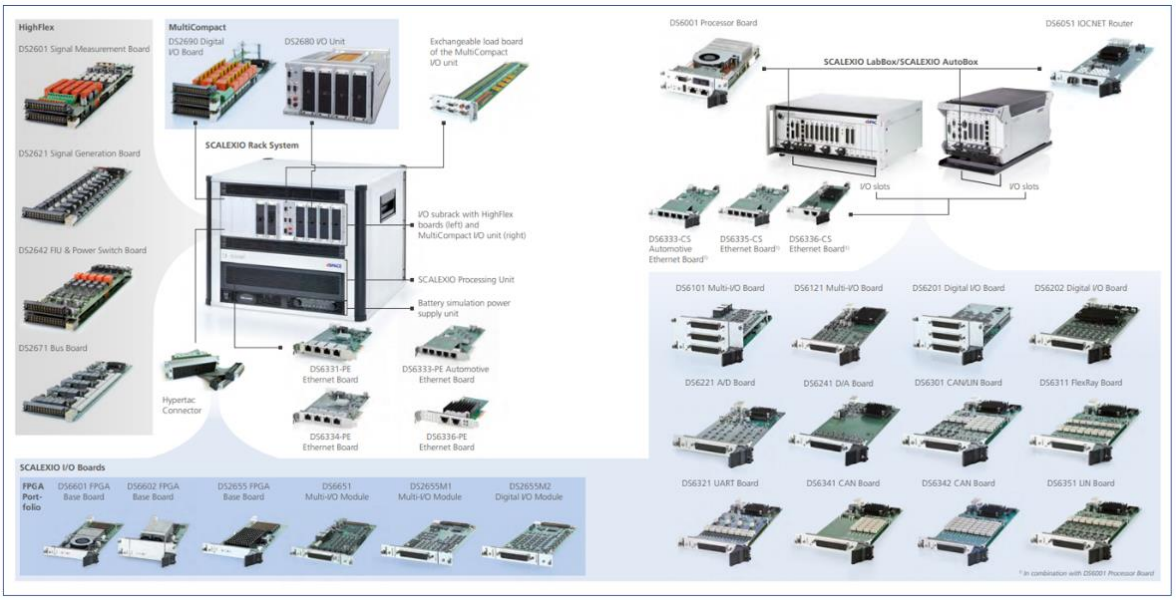


Fig. 6 SCALEXIO Modules Board

FPGA

An FPGA is an integrated circuit, and it can be programmed and configured even after manufacturing. This feature makes those types of components very versatile, and compatible with most application that has high computation requirements. The FPGA configuration is performed through a specific type of language, the hardware description language (HDL).

The name stands for field-programmable gate array, and it delights the primary advantage, which is the presence of an array of programmable logic blocks whose connection can be reconfigured allowing to wire them together. Depending on this configuration, blocks can act as simple logic operator al AND and OR or perform more complex functions. This re-programmability can be performed through downloadable software even after the integration into an application.

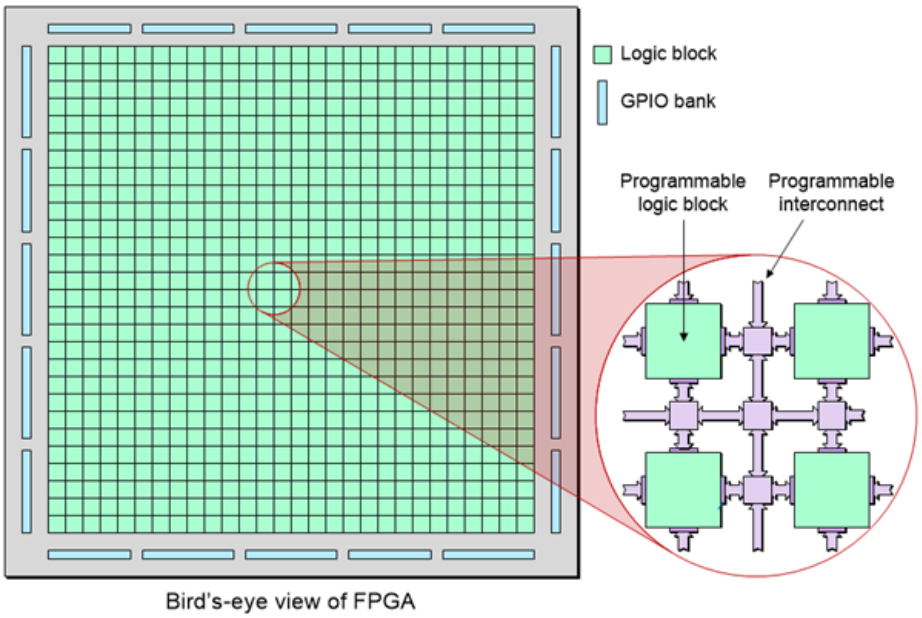


Fig. 7 FPGA Architecture

Another great advantage comes from its architecture design “sea of gate”, FPGAs are capable of parallel data processing, which means operations are processed at the same time in parallel and not sequentially. This type of architecture makes FPGAs suitable for most high-performance requirement applications.

Electrical Motors

The electric motor (EM) is a fundamental component in almost every sector of the industry, it transforms electrical energy into mechanical energy through the force generated by the magnetic field and electric current as explained by the Lorentz law. EMs are divided into two main categories based on how they are powered, direct current (DC) motors and alternating current (AC) motors.

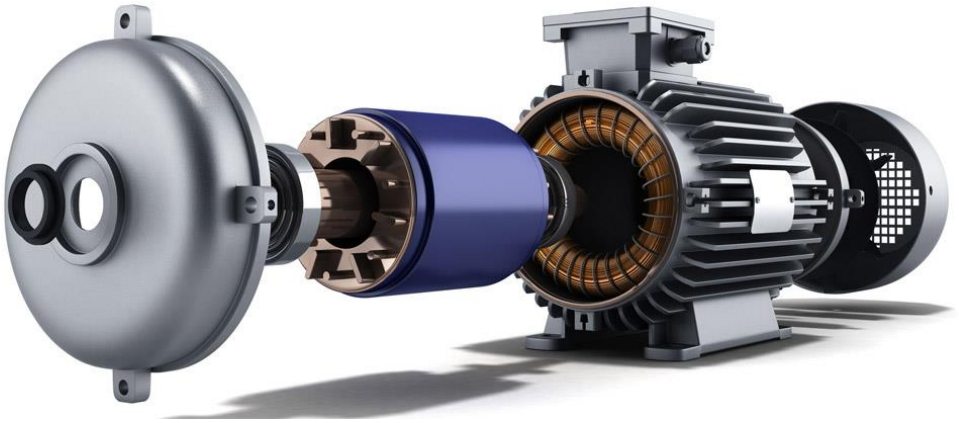


Fig. 8 Electric Motor

Generally speaking, there are two main components always present no matter what type of motor, a stator which often is also the structure of the machine, and a rotor which generate the mechanical movement. Depending on the type of motor, we have different characteristics that make a motor more suitable than others for any kind of job; characteristic as costs, durability, maintenance, and physical dimension. And other that depends on the parameters and configuration of the motor, such as speed, torque, efficiency, etc…

One of the biggest differences between an internal combustion engine (ICE) and an EM is the starting conditions. The ICE at low rotations per minute (rpm) develops very low torque, that's why it is needed a clutch to make it operate at higher rpm. The EM instead generate the maximum torque already at the starting point and remain constant as the motor power increase till the base speed, where the power reaches its maximum value, and the torque starts to decrease.

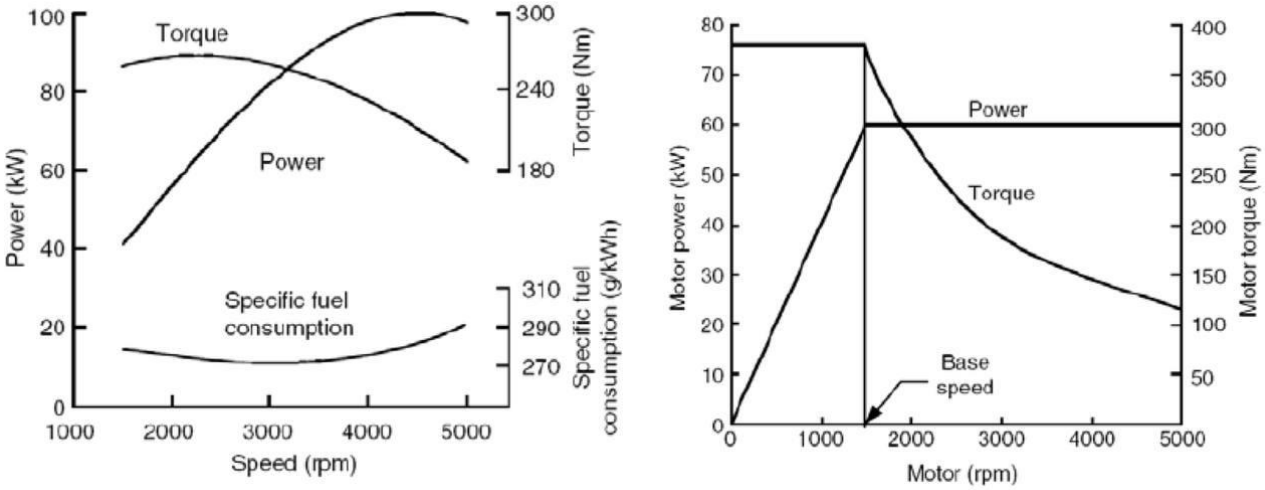


Fig. 9 Power-Speed-Torque Diagrams

Another important feature and advantage of the Ems for automotive applications is the possibility to use it as a generator. This means that the torque is negative and is generated by the vehicle for example when braking or during descent, in this case, the motor is not consuming energy but is generating it, and it can be used for recharging the battery.

DC Motors

DC motors are easy to power and control, but not commonly used in automotive applications, that's because they need to be connected to a commutator. The commutator allows each armature coil to be energized in turn and connects the rotating coils with the external power supply through brushes. This component limits those type of motors for multiple reasons:

- Axial overall dimensions
- Wear, spark, and maintenance
- Speed limit
- Cooling difficulties
- Difficulty with watertight constructions

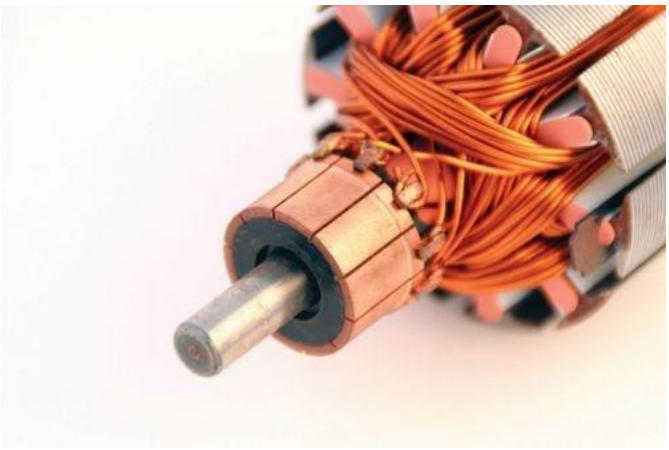


Fig. 10 DC Brushed Motor shaft

A possible solution for this problem comes from the brushless motor, which uses one or more permanent magnets, and thanks to a different configuration it avoids the presence of the commutator and the brush. The advantages of brushless respect to traditional brushed motors include a long-life span, little or no maintenance, and high efficiency. Disadvantages include high initial cost and more complicated motor speed controllers.

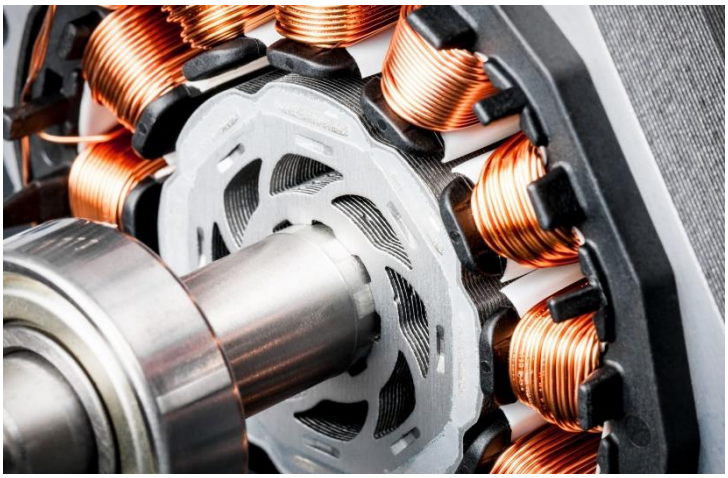


Fig. 11 DC Brushless motor

AC Motors

Driven by alternating current, AC motors are characterised by only two basic parts: an external stator, the one connected to the power source and thar generate a magnetic field, and an internal rotor that produces another magnetic field, this time rotating, that create the mechanical movement. In automotive, these type of motors finds a larger use than its counterpart in DC, thanks to various advantages:

- Less overall volume
- Robustness
- High speed
- High efficiency
- Liquid cooling
- Watertight construction
- Silent operation

As a downside, their calibration can be a bit more complex.

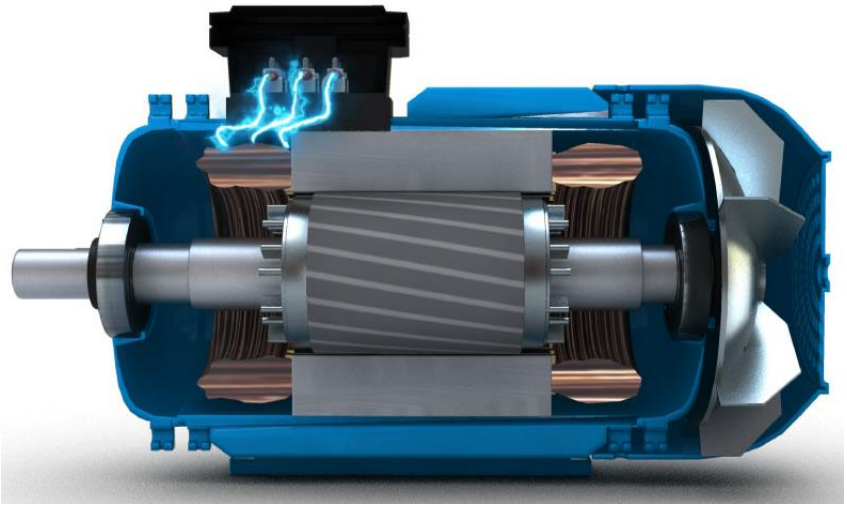


Fig. 12 AC Motor

To control the output speed ($\omega_s\ rad/s\ and\ N_s\ RPM$) of these motors there are two methods, change the numbers of poles pairs (P) and change the input source frequency (f).

$$\omega_s = 2\pi\frac{f}{P} \qquad N_s = 60\frac{f}{P}$$

Obviously changing the number of poles is an option only during the design of the machine, before the manufacturing. Then the frequency is the only parameter that can be changed in order to change the speed.

AC Motors can be divided into two main categories, Synchronous and Asynchronous motors depending on the characteristics of their rotating magnetic fields

AC Synchronous Machine

A synchronous motor's rotor normally rotates at the same speed as the magnetic field of the stator. The machine is composed of a stator that contains multiphase AC electromagnets, connected to the power source whose frequency determines the rotation speed of the generated magnetic field, and a rotor with permanent magnets or electromagnets that turns at the same rate as the stator's field, and that generates a second synchronised field as well as the mechanical power.

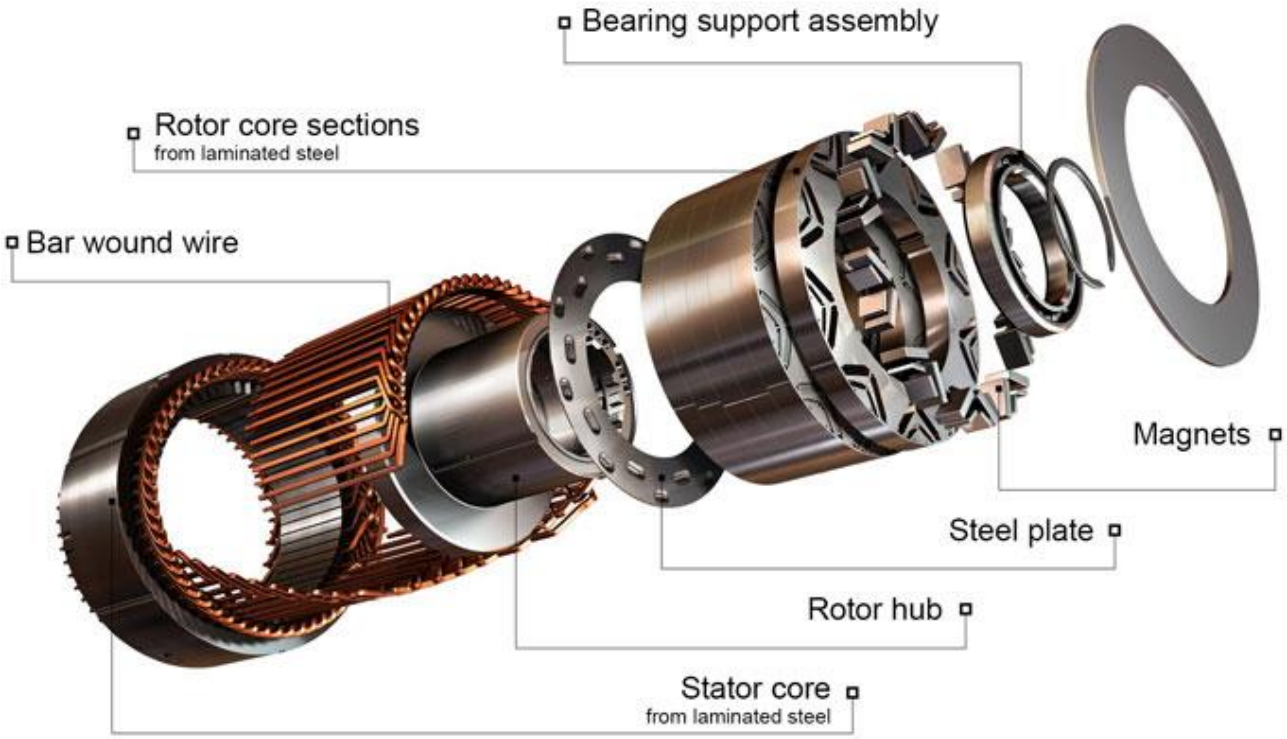


Fig. 13 AC Synchronous motor with permanent magnets

Synchronous motors are very versatile and can adapt to various applications, from self-exited sub-fractional horsepower sizes to high-power industrial sizes. The first is perfect where precise constant speed is required and low torque is needed, while the second is used in high-power industrial applications for their efficiency.

AC Asynchronous motor

In this section, it will be discussed the Asynchronous motor, precisely the Squirrel cage induction motor (SCIM) with a more in-depth view with respect to the precedent motors because it's the motor analysed and modelled in the thesis project.

These types of motors are widely utilized in industrial applications because they are self-starting, economical and require little maintenance.

There are two types of induction motors:

- **Wound type**
  Rotor circuit is composed by three-phase windings similar to that of the stator and are connected to each other through slip ring to an external resistance in star configuration.

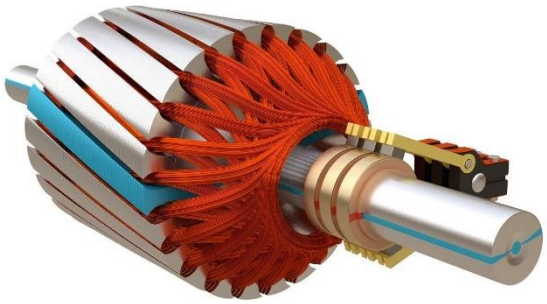


Fig. 14 Wound rotor

- **Squirrel cage type**
  Rotors consist of a cylinder of aluminium or copper lamination short-circuited to each other.

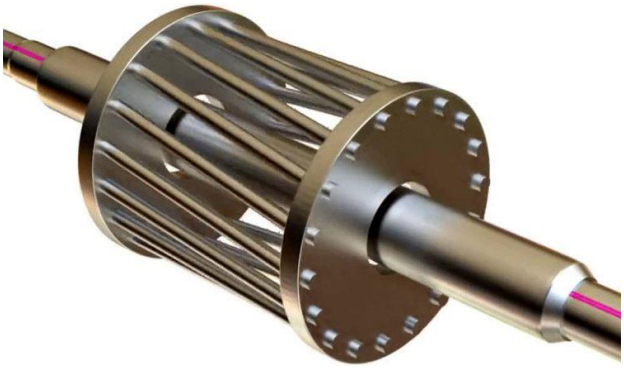


Fig. 15 Squirrel cage rotor

As in the synchronous version, the power is supplied through the motor's stator and which generates the rotating magnetic field, but in an induction motor, the torque is generated based on the force created by rotor currents, which are induced by the rotating airgap field generated by the stator currents. Therefore, there's no need for an electrical connection with the rotor.

When supplied with three-phase currents the stator generates a rotating electromagnetic field (EMF) in the airgap between the stator and rotor, which rotate in synchronous speed ω_s. This time-changing magnetic field induces voltages on rotor bars, as defined by Faraday's law. Then, due to the rotor's short-circuited conductors, an induced current generates a force following Lorentz's force law.

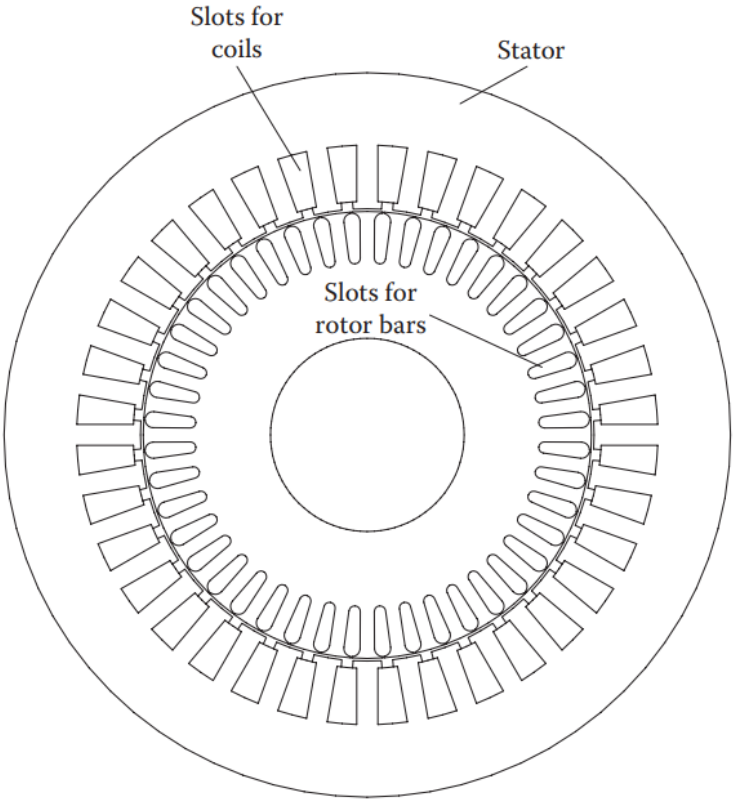


Fig. 16 Section of a Squirrel cage IM

The characteristic of the asynchronous motor is that to generate the current, voltage, and hence torque, the rotor's magnetic field and the rotor itself must rotate slower than the stator's magnetic field. The difference between the two velocities is called slip.

When a load is present, the speed of the rotor drops, this increases the slip, induces a magnetic field, and generates the torque necessary to move the load.

As reported previously, the synchronous speed is:

$$\omega_s = 2\pi \frac{f_s}{P} \qquad N_s = 60 \frac{f_s}{P}$$

Where f is the frequency and P the pole pairs

Slip can vary from 0 where the torque is null to 1, where the rotor is stalled. It can be obtained from the synchronous speed and the actual rotation speed of the rotor N_r defined as:

$$s = \frac{\omega_s - \omega_r}{\omega_s}$$

Knowing the slip, the actual frequency of the rotor's magnetic field (f_s) and rotational speed (ω_s) can be found:

$$f_r = sf_s \qquad \omega_r = s\omega_s$$

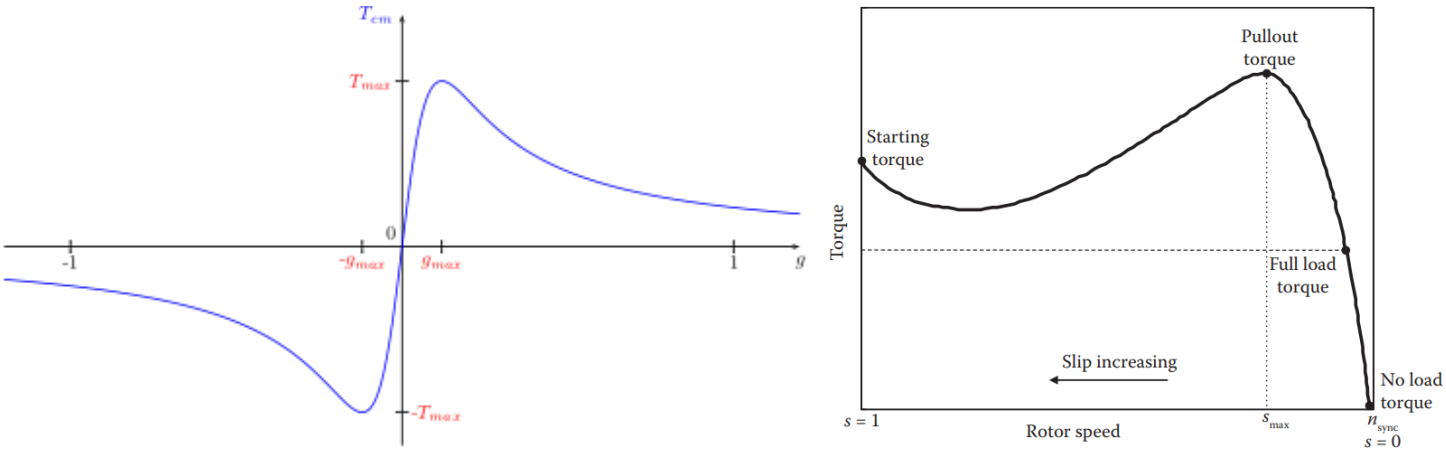


Fig. 17 Slip characteristic.

In the figure above, is described the slip characteristic with the slip on the x-axis, and the torque on the y-axis.

- $\boldsymbol{s = 1}$
  This is the initial condition where the rotor rotational speed is 0. Here there's the starting torque.

- $\boldsymbol{0 < s < s_{max}}$
  In this range, there's the typical operational point where the torque keeps increasing proportionally with the slip until reaching its maximum value at s_{max}, at s = 5% there's the typical full load torque.

- $\boldsymbol{s = 0}$
  Here the rotor's speed ω_r has reached the synchronous speed, hence no volt and current are induced, and consequently, no torque is generated.

- $\boldsymbol{s < 0}$
  When the slip is negative, it means that the rotor speed ω_r is greater than the stator speed and that the machine is acting as a generator, producing energy.

As the load keep increasing, the slip follows it proportionally. If the slip exceeds the maximum value s_{max}, the induced torque starts to reduce. This is because the power factor decreases with a higher rate than the increase in rotor current due to the high rotor reactance caused by the higher rotor frequency. This causes the torque required increase beyond the pullout torque, consequently the rotor starts to lose velocity and finally stall.

Its operational principle is similar to that of transformers, with the difference that is present an airgap in the magnetic circuit that increases the level of magnetizing current. Also, the frequency of the induced EMFs in the rotor is lower than in the stator. So, when the rotor is stationary, the machine act as a transformer with an airgap

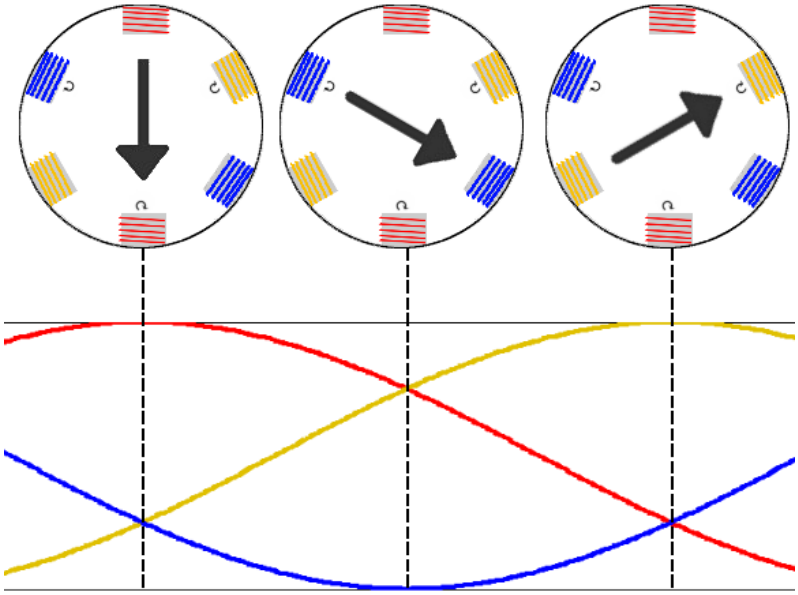


Fig. 18 Three-Phase Currents

In a three-phase, one pole-pairs induction motor, all the three windings of the stator are fed with sinusoidal AC current with frequency ω and 120° of phase shift between each other. The three currents I_{abc} generates alternative magnetomotive forces (mmfs) $F_a, F_b \; and \; F_c$

$$F_a = F_a sin(\omega t)$$

$$F_b = F_b sin(\omega t \; + \; 120°)$$

$$F_c = F_c sin(\omega t \; + \; 240°)$$

The resultant mmf F_s of the stator ca be expressed as:

$$F_s \; = \; F_a e^{i0°} + F_b e^{j120°} + \; F_c e^{j240°} \; = \frac{2}{3} F_s e^{(\omega t \, - 90°)}$$

As it can be noticed the resultant mmf has frequency ω equal to the frequency of the supplied current. Then the interaction between the stator mmf and the rotor conductors, as the Faraday law say, induces a voltage and consequently a current, that then will produce the torque.

Inverter

In recent years the introduction of power electronic and electrical propulsion in automotive applications plays a new main role in the need of increasing engine performance, comfort and safety. All these new applications have a high-power requirement that needs efficient management, such management and modulation functions are feasible by power electronics devices. In electrical vehicles (EV) and hybrid electric vehicles (HEV), the electrical and electronic accessories are supported by the battery. This power is modulated by using the converters. Converters use power electric components (diodes, transistors, etc..) to obtain the desired output:

- Rectifier, AC to DC conversion
- Inverter, DC to AC conversion
- Chopper, DC to DC conversion
- AC to AC converter

Fig. 19 Inverter by Bosch for automotive applications

To make it more comparable to a real system, it has been decided to models and simulates an inverter on Simulink, due to the high frequency required to achieve better precision and performance, it has also been decided to model it in FPGA. Due to its complexity, the inverter makes

the model behaviour to be more real, instead of the first ideal model where the controller directly drives the motor, adding delay and imprecision in the signal.

PWM in three-phase inverter

In vehicular applications, the amplitude of the output, as well as its frequency, is used to control the torque and speed of the motor. To generate both amplitude and frequency variable outputs, a PWM can be used. The PWM (or Pulse-width modulation) are generated by comparing a given reference signal (V_{ref}) to a carrier ($V_{carrier}$) to generate a square wave driving signal for the switches following the next relation:

$$\begin{cases} u_i = 1 & if \quad V_{ref} > V_{carrier} \\ u_i = 0 & if \quad V_{ref} < V_{carrier} \end{cases}$$

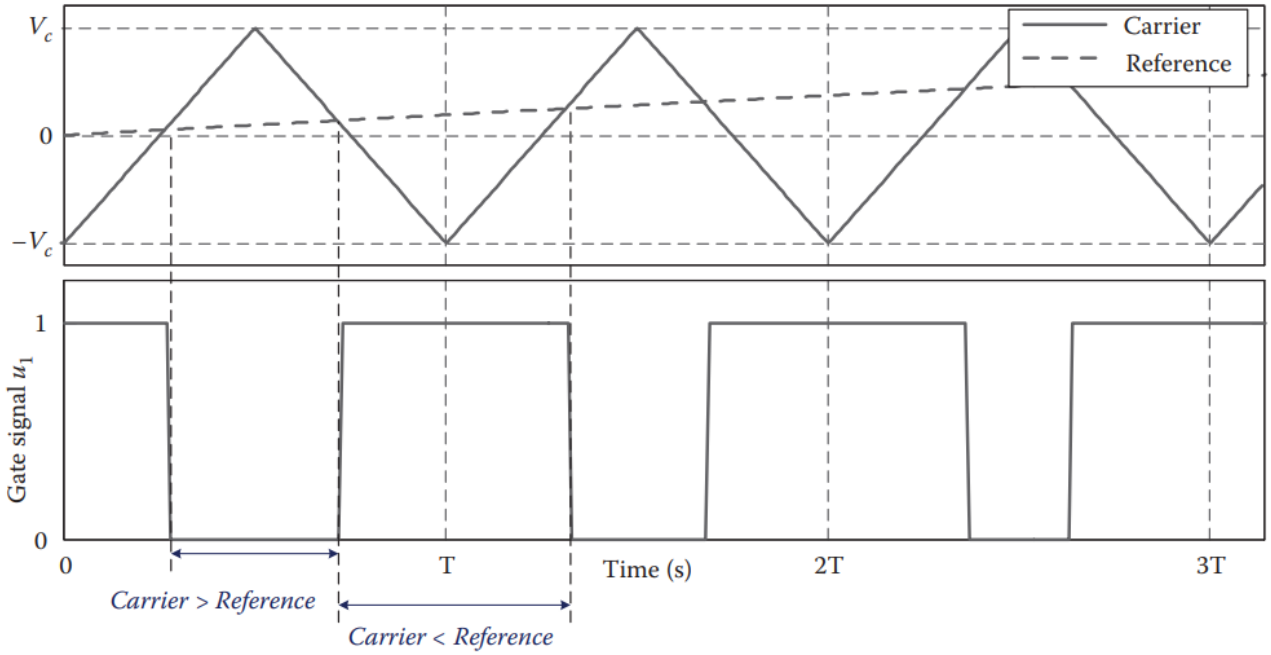


Fig. 20 PWM Generation

And here we can see the output of a sinusoid reference signal:

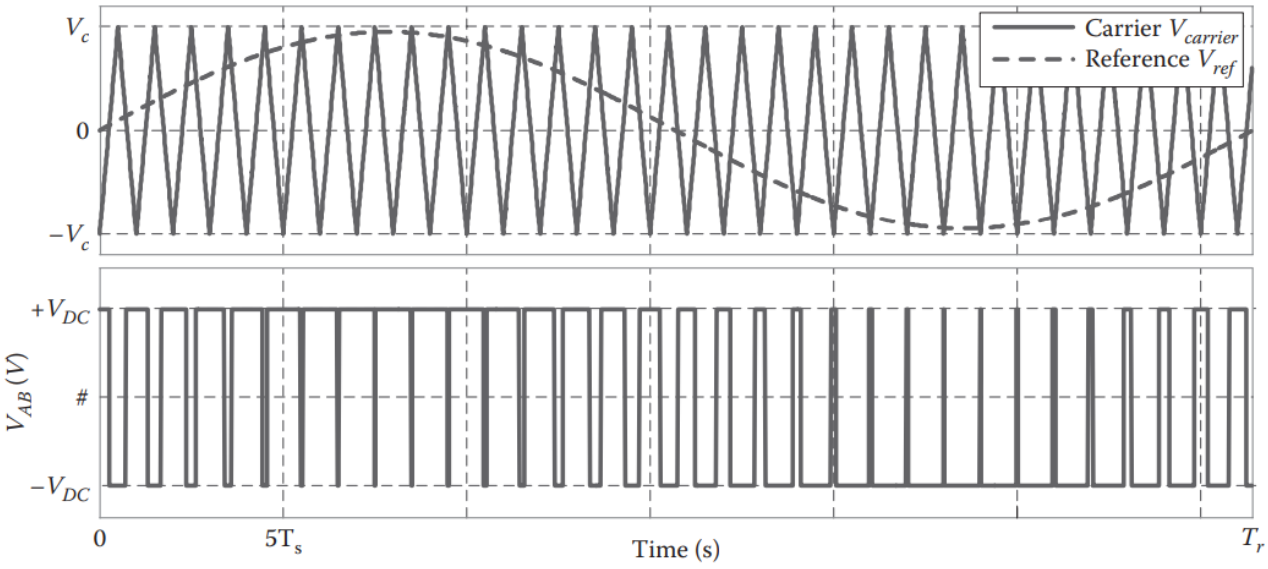


Fig. 21: PWM with sinusoid input

So, for a three-phase inverter, there will be three PWM signals, generated from V_a , V_b and V_c, that will drive the switches of the inverter. To be more precise the inverter needs six PWM signals, one for each switch, but as shown in Fig.22 the couple of switches of the same row cannot close at the same time so the two PWM signals are one the reciprocal of the other.

DC to AC Three-Phase Inverter

The DC to three-phase AC power conversion is performed by the three-phase inverter. The two main applications of this type of inverter are high-power grid-tie inverters for electrical energy transportation and a three-phase ac drive system. This electronic unit is very important in any transportation applications as almost every motor used for traction (automotive, railway, ship) are three-phase ac.

Electrical circuit

The three-phase inverter electrical circuit as shown in Fig.22 is composed of three-row in parallel to the direct current voltage source, usually the battery. Each of these rows is composed of 2 bidirectional switches such as Mosfet or IGBT with an antiparallel diode configured so that they cannot close simultaneously, to prevent short-circuiting the battery. The middle point of each leg is

also one of the outputs of the inverter in automotive motor drive, the three phases are connected in Y configuration.

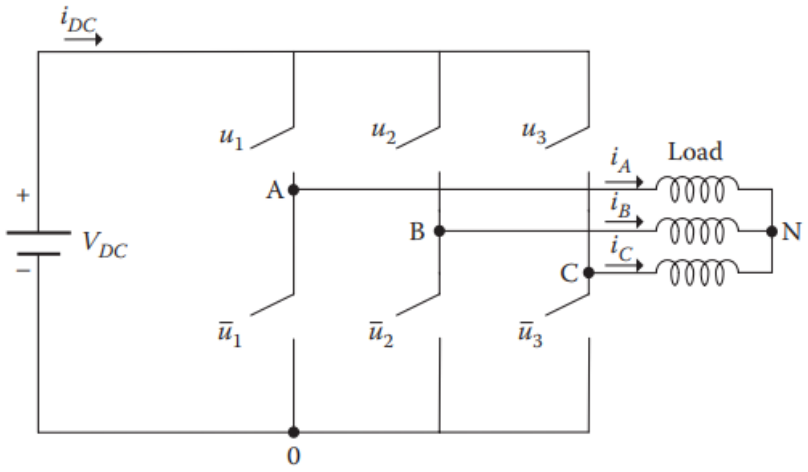


Fig. 22: inverter electrical circuit

Three-phase RC Inverter

Due to some problem that occurred while testing the tabular inverter it has been decided to switch to a simpler RC Inverter. The RC inverter is a simpler device not used in automotive applications, but for the simulation purpose of this thesis, even with some imprecision, works just fine. These types of electric nets are dynamic systems varying depending on time and are composed of resistors, condensers, and inductors.

Electrical circuit

The electrical circuit of a three-phase RC inverter is composed of simple components such as resistance R and capacitance C whose values are constant.

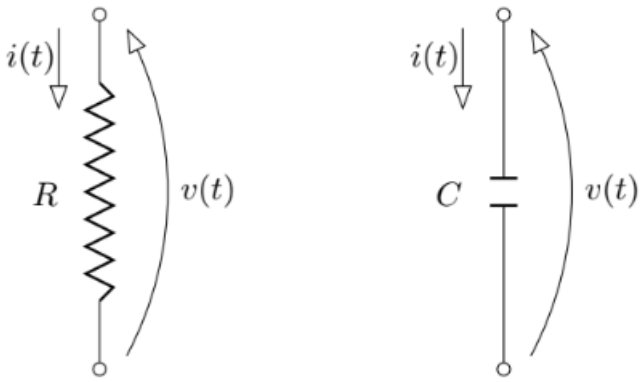


Fig. 23: Resistor (Left) Condenser (Right)

The resistor is characterized by a voltage at the terminal extremities which is proportional to the current flowing inside the component itself. Its elementary law is:

$$V(t) = Ri(t)$$

The capacitor is characterized by a current that flows inside the component that is proportional to the derivative of the voltage at the terminals of the component itself. We therefore have:

$$i(t) = C\frac{d}{dt}v(t)$$

In general, from the physical point of view, the state of a system is represented by the properties which define the potential energy that is stored in an electrical circuit; in the circuits consisting of resistors and capacitors the energy is stored in the condensers ($e_{capacitor}(t) = \frac{1}{2}Cv(t)^2$) and if present in the inductors ($e_{inductor}(t) = \frac{1}{2}Li(t)^2$). The state equation can be obtained easily starting from the previous equation and applying the first and second Kirchhoff laws which respectively say:

- The sums of the currents entering a node are zero (Kirchhoff Current Law - KCL)
- The sums of the tension along a loop are zero (Kirchhoff Voltage Law - KVL).

As the state variables are related to capacitors (which represent the system's reactive element), the system's order depends only on the number of such elements present in the network.

RC Integrator

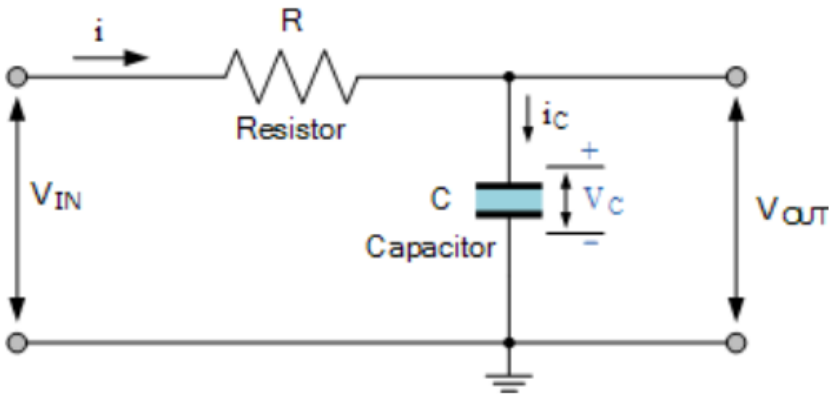


Fig. 24: RC Integrator circuit

The figure above represents a basic resistor-capacitor (RC) circuit in series, this circuit has numerous uses and applications, from simple charging/discharging to high-order filter circuits. Even though it looks very simple, it can be obtained various outputs by changing the type and frequency of the input. Also known as the RC integrator circuit, the input is connected to the resistance, while the output is taken across the capacitor which charges up when the input is high and discharges when is low. This behaviour comes from the reactance of the capacitor X_c which increases at the decrease of the frequency and vice-versa following the formula:

$$X_c = \frac{1}{2\,\pi\,f\,C}$$

Knowing that the capacitor is a frequency-dependent element, the level of charge reached is equal to the time domain integral of the current. That is because the real capacitor does not charge instantaneously, but exponentially, so it takes a certain amount of time to fully charge.

$$i_c(t) = C\frac{dV_C(t)}{dt}$$

The rate at which the capacitor charge is proportional to the time constant, equal to the product of the amount of resistance R and capacitance C.

$$\tau = RC$$

It is important to notice that this time constant corresponds to the time required for the component to reach 63.2% of the maximum voltage and 36.8% when discharging from the maximum voltage.

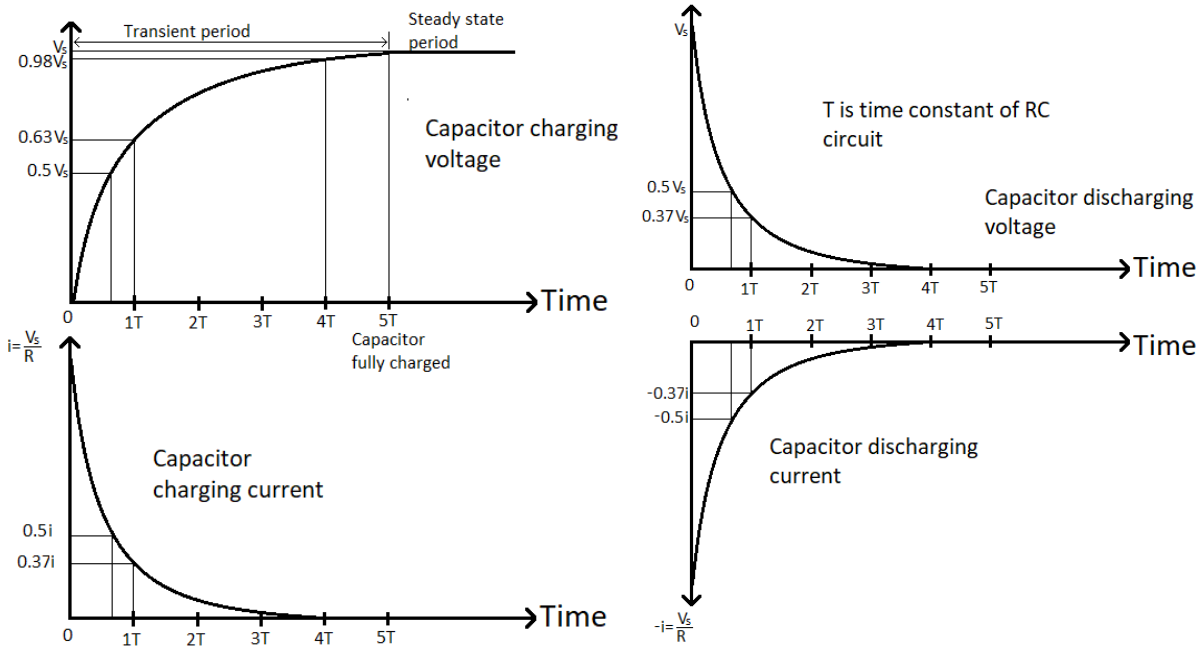


Fig. 25 Capacitor charging and discharging voltage and current.

Voltage is connected to the resistor, the current flowing in the circuit can be obtained by applying the KCL to the RC series obtaining:

$$i(t) = \frac{V_{in}}{R} = \frac{V_R}{R} = C\frac{dV}{dt}$$

Therefore, substituting and rearranging to obtain V_{out}:

$$V_{out} = V_C = \frac{Q}{C} = \frac{\int idt}{C} = \frac{1}{RC}\int V_{in}\,dt$$

To summarize, the output voltage V_{out} of an RC integrator circuit corresponds to the time integral of the input voltage V_{in} divided by a constant value RC, which represents the time constant, τ.

$$V_{out} = \frac{1}{RC}\int V_{in}\,dt$$

Electronic Control Unit

Commonly called Controller, the electronic control unit (ECU) is the heart of every electronic system. The increasing complexity and requirement for safety need a more efficient, precise, and accurate control of the systems, and that's why some modern vehicle has over one hundred different ECUs. This component uses a microprocessor to process inputs coming from sensors or by estimating them. Through control algorithms they can predict the behaviour of a system, comparing it to the desired one and adjusting it if needed.

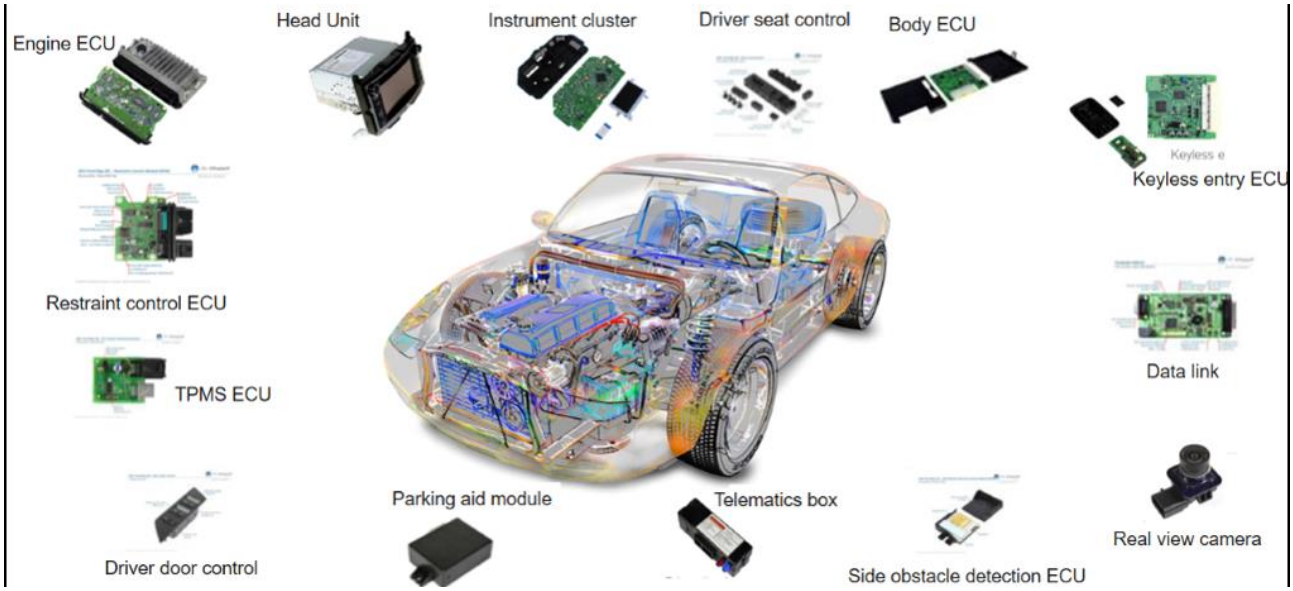


Fig. 26 Various ECUs in a vehicle

One of the most important ECU is the engine control unit or engine control module (ECM), responsible for controlling the engine and every component related to it.

As said, the thesis' goal is to reproduce a virtual plant composed of an inverter and an induction motor to test a real component, the ECM. Unfortunately, we didn't have access to any of those, so as a side activity of the thesis we also simulate its behaviour, but still making the model able to accept an external input to connect real hardware.

There are different types of control algorithms that depend on different characteristics, the first big division can be made from Open Loop and Closed Loop systems.

Open Loop Control

In this type of control system, there is no feedback from the output, which means that the output does not affect the behaviour of the control system. It's very simple, is quicker to perform operations, and is inexpensive, but the accuracy is low and for this reason, it cannot be used in safety-related application.

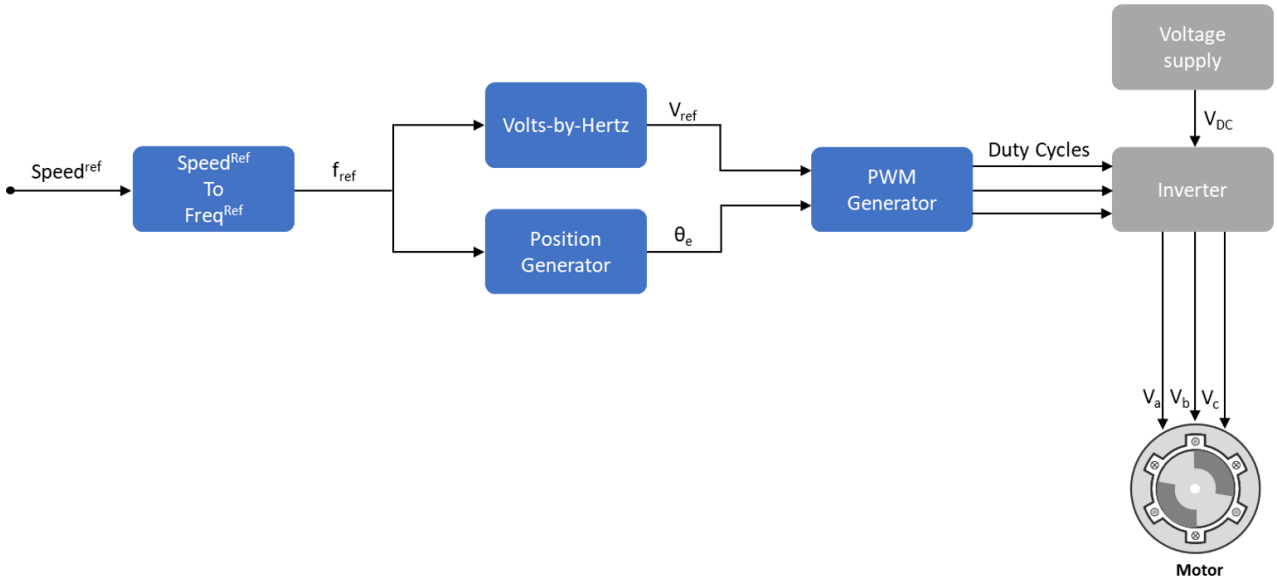


Fig. 27 Open Loop System

Recently new research and technology made this type of control valuable also for safety-related applications since estimating the feedback became more precise and reliable. An example is a speed-sensorless control for an induction motor. Open-loop estimator arises a lot of interest because of their simplicity and low-cost profile even maintaining the accurateness needed for this application.

Closed Loop Control

In closed-loop control systems, the output affects the behaviour of the control. The feedback is used to tune the input, to reach the desired output. These types of control systems are more complex, expensive, and slower than the previous ones, but are very precise and accurate making them perfect for safety-related applications.

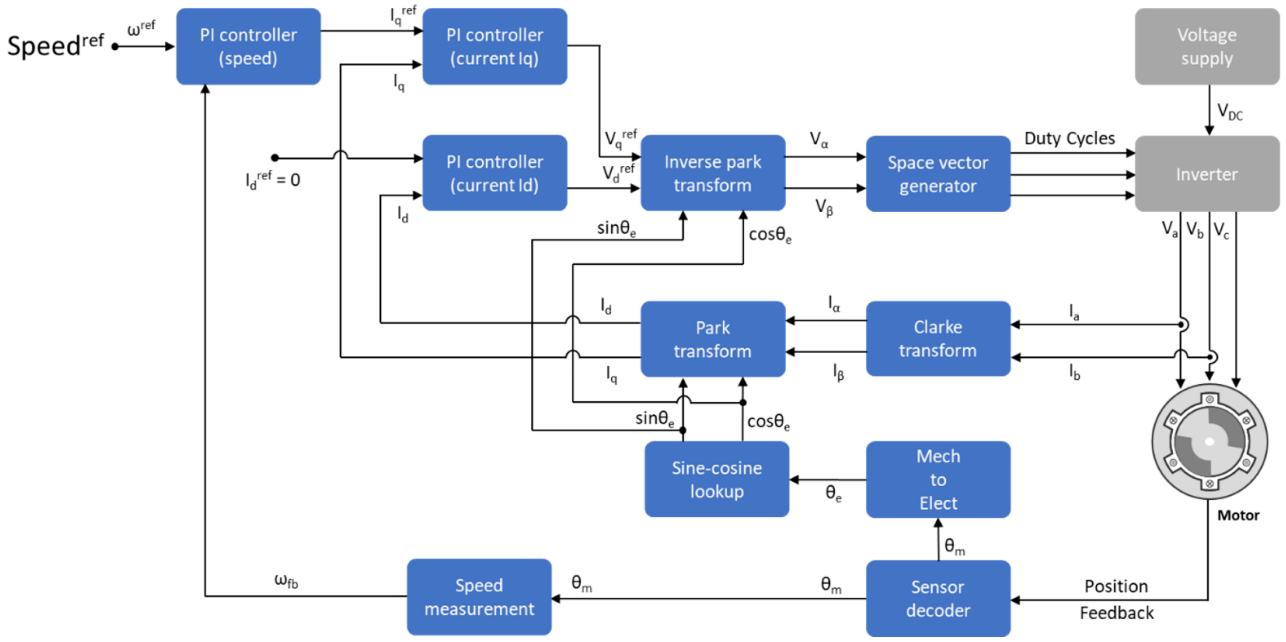


Fig. 28 Open Loop system example

Taking as an example the ECM of an electrical machine, from a general view, one of the main controls that it must perform is on the speed (RPM) and o the torque (N/m). Either sensored or sensorless ECUs must provide a certain level of precision in the response, to assure safety and comfort.

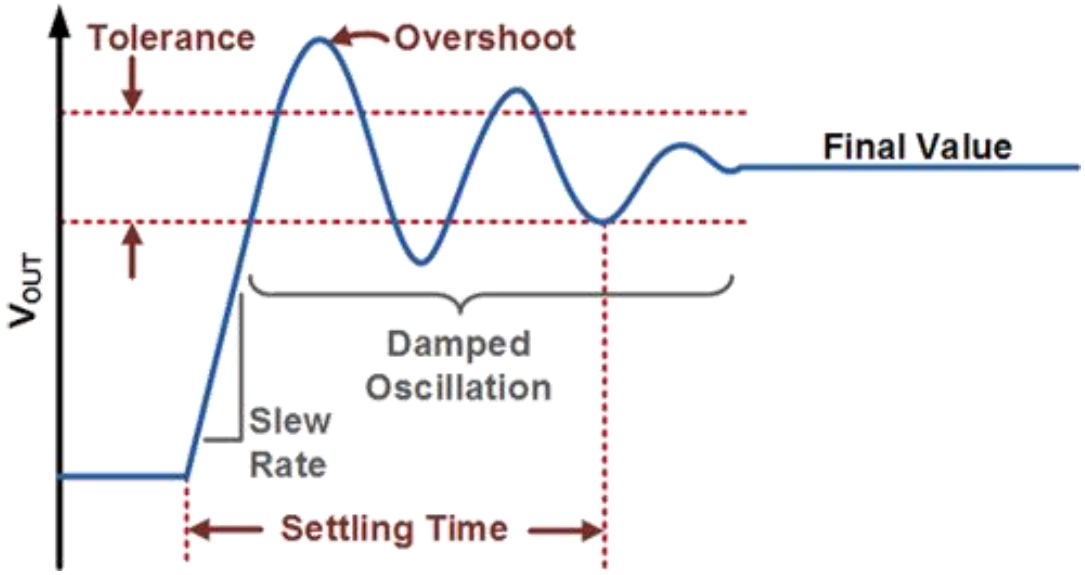


Fig. 29Typical response of a system

The graph above represents a typical response from a system.

The information that can be extract from the graph are:

- **Slew Rate:** the slope of the initial increase.
- **Rise Time:** the time needed for reaching the desired value.
- **Overshoot:** the highest value reached
- **Undershoot:** the lowest value reached
- **Tolerance:** the range around the final value which is acceptable
- **Settling Time:** time to stabilize into the tolerance.

Let's for example consider the graphic above as the change of the speed of the motor over time and let's analyse the characteristic. The first important requirement is that the target velocity is reached, and at the right time, if it is reached too early it means that the torque reached is too high, this can result in an uncomfortable acceleration, as well as a waste of energy. Another waste is surely high overshoot and settling time.

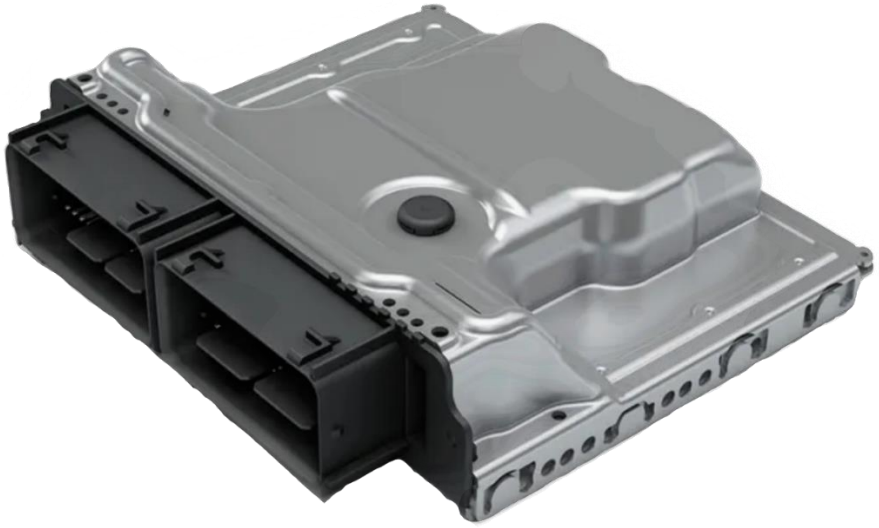


Fig. 30 Typical Engine Control Unit for vehicle

To ensure these requirements, the ECM need to be calibrated. Calibrating these devices means finding the right control parameters, this operation is quite difficult and time taking.

The control algorithm chosen for the thesis is the field-oriented control (FOC). The FOC is a variable-frequency drive (VDF) which consists of the visualization of the three-phase current of the stator in a rotating reference frame with two components direct (d) and quadrature (q), one defining the magnetic flux and the other the torque. Three modes of control can be used:

- **Torque Control:** this is the most used for traction applications, it uses a reference torque to regulate the output
- **Speed Control:** this uses a speed reference value from which it generates a torque reference for torque control
- **Position Control:** this is the less common mode, and it uses the position of the motor

The FOC is a closed-loop algorithm that requires real-time feedback from the motor which are stator currents and rotor position. Some controllers can estimate those parameters and work in an open-loop configuration

Clarke & Park Transformation

As previously said the algorithm control used is the FOC which uses a rotating dq reference frame. In order to change from the three-phase fixed frame of the currents to the desired frame, the Clarke and Park Transform will be used.

The three-phase frame is used for the representation of the three current of the stator (I_{abc})

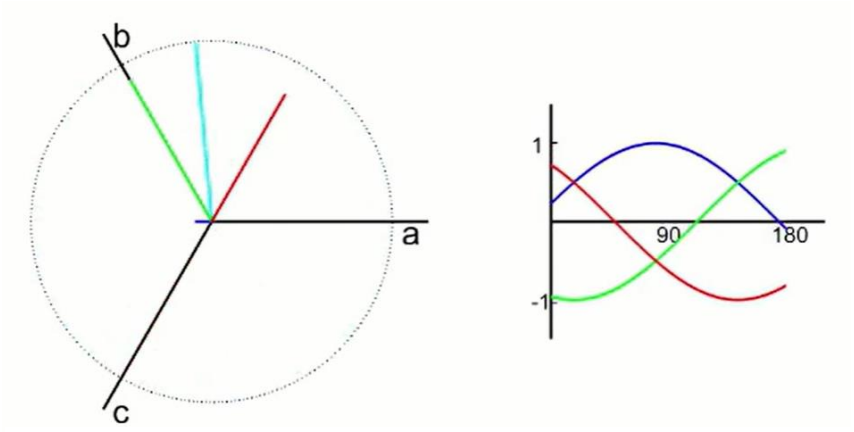


Fig. 31 a b c frame

Clarke is the first passage; it converts the components of a fixed three-phase frame (a b c frame) into two components still in fixed frame (α β frame)

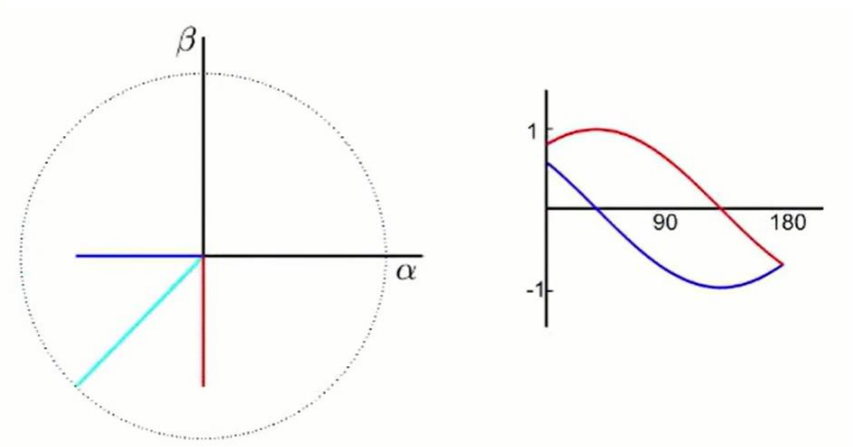


Fig. 32 α β frame

Park is the next transform the two components of the α β frame in a rotational orthogonal frame (d q frame)

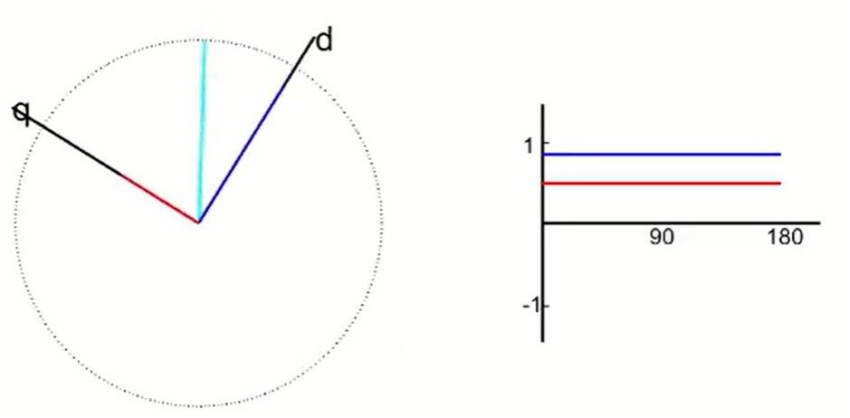


Fig. 33 d q frame

The mathematical passages are shown:

- **Clarke**

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

- **Park**

$$i_{dq} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}$$

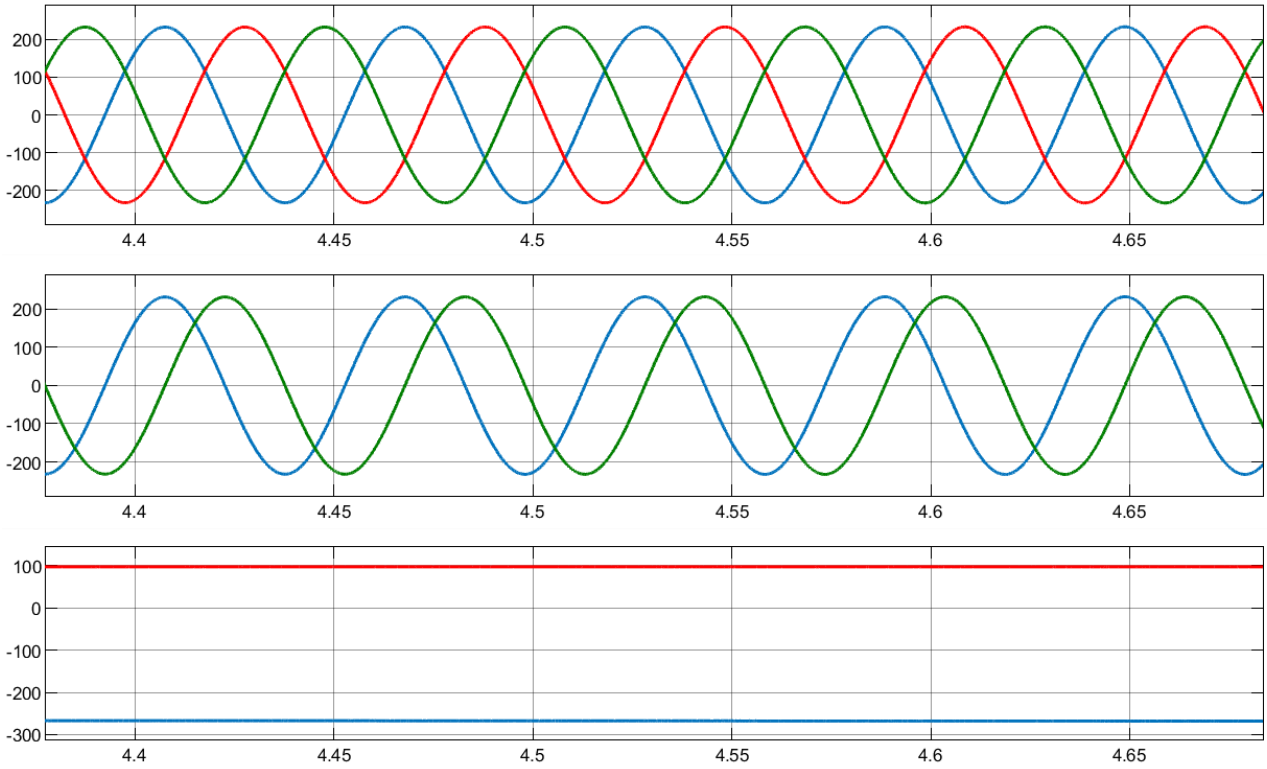
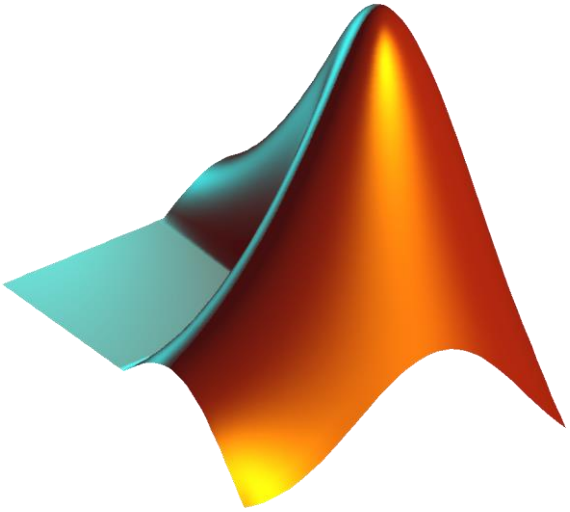


Fig. 34 Clarke & Park Transform

Toolchain

Here it will be given a brief introduction to the software and hardware used for this project.

MATLAB & Simulink

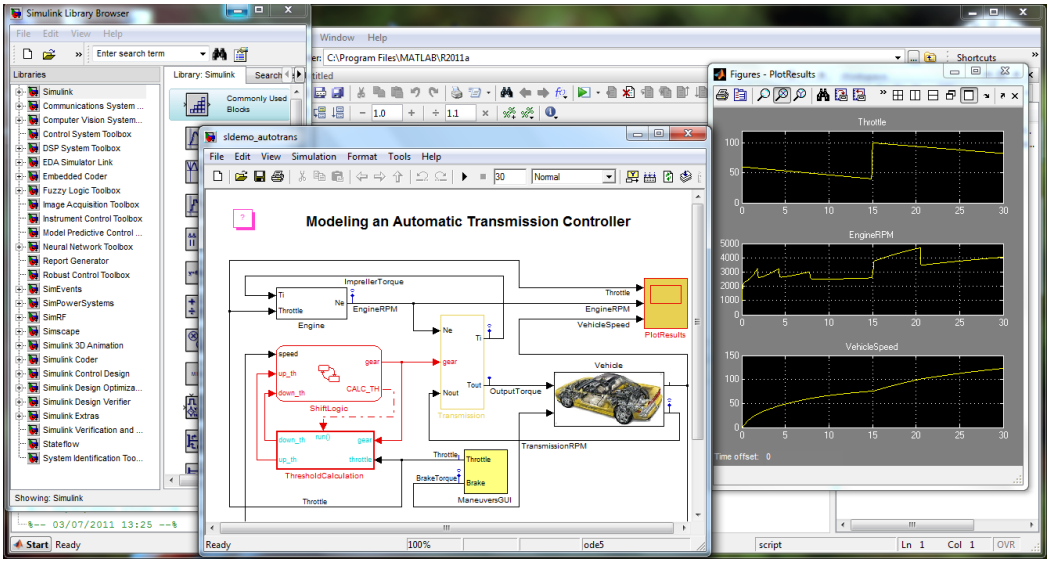


MATLAB or Matrix Laboratory is a programming language able to perform matrix manipulation, design of an algorithm, communication with other programs and languages, draw functions and data and development of graphic user interface.

It offers numerous optional toolboxes, which make MATLAB suitable for different applications in every field of engineering, science, and economics.

Simulink is the toolbox utilized for the implementation of all the models created for the thesis.

Simulink is a software integrated into MATLAB used for modelling, simulating, and analysing dynamical systems. This is a Model-based graphical programming environment where the user places block diagrams to project, emulate, and test, before switching to the hardware.



System Generator for DSP (XSG)

Xilinx system generator is a DSP design tool created for the MATLAB addon Simulink focused on the FPGA design. It's a library that adds in Simulink the Xilinx-specific blockset that uses the friendly Simulink environment of model-based design to simplify the FPGAs or RTL design methodologies previously needed, even without knowing anything about FPGA programming languages, like HDL or VHDL.

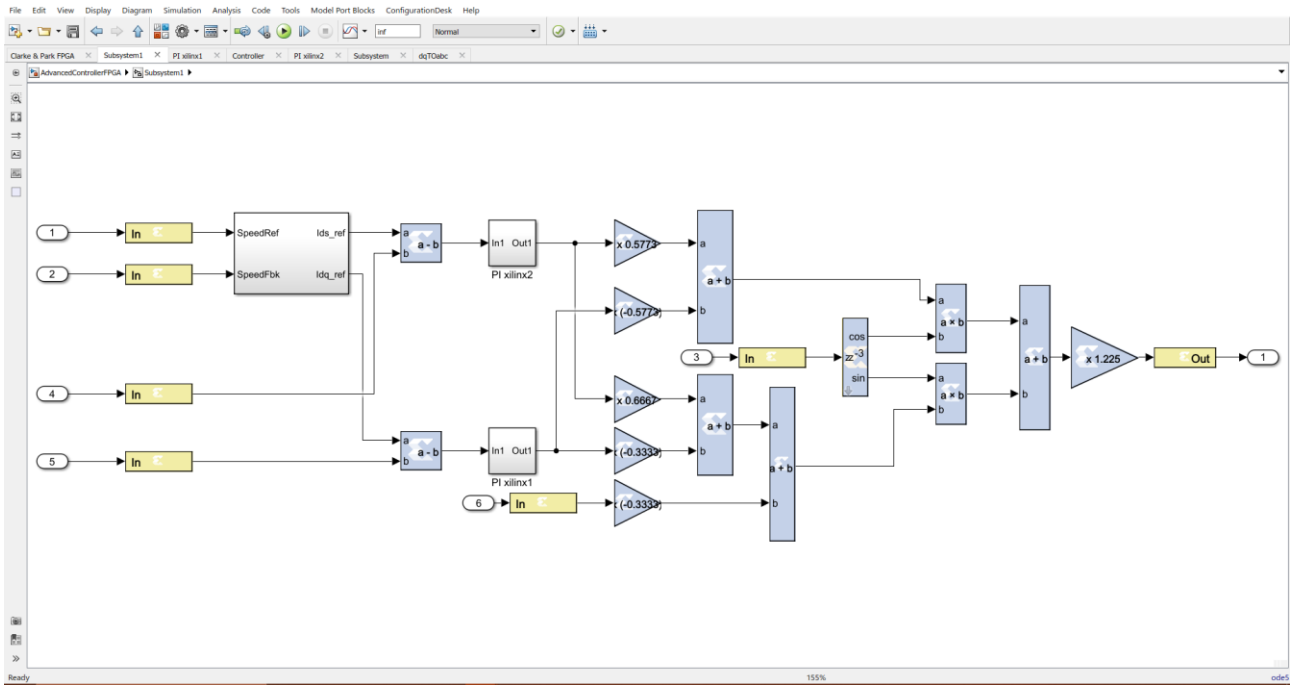


Fig. 35 Xilinx Blockset

The library contains all the basic operations like sum, multiplication, sine, cosine etc… Instead, for a more complex function like an integrator, we had to build it by ourselves. Thanks to this tool it was possible not only to design all the models needed for the project but also to test them and verify their correctness on a pc before uploading them on the FPGA.

dSPACE Software

To prepare the models to make them work on the HIL simulator, a procedure must be performed. First, in MATLAB the model is separated into two distinct subsystems, the processor, and the FPGA, this separates the code that later will be uploaded on the processor and the FPGA. Finally, a build will be created to export the code generated from MATLAB to Configuration desk.

Configuration Desk

Configuration desk is a graphic tool for the implementation and configuration of the model created on MATLAB to the real-time simulator from dSPACE like Scalexio. This tool is able to configure all the I/O functions for communication with an external device under test (DUT).

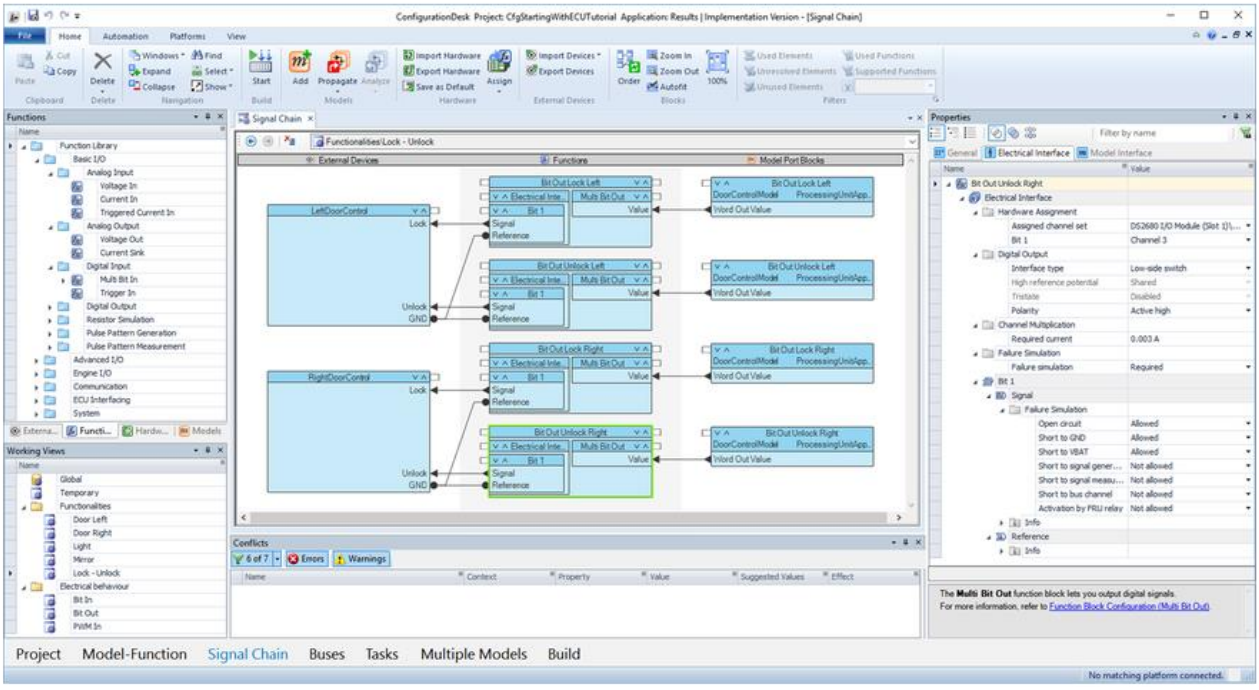


Fig. 36 Configuration Desk

After the configuration is done, it can be done the final build that will create the file to be uploaded on the real-time application SCALEXIO, using Control Desk

Control Desk

Control desk manages the graphic interface when using the simulator. This tool can perform numerous operations, from the calibration of the ECM, and access to the bus system like CAN, to the test itself.

Fig. 37 Control Desk

It provides the tool to link all the tunable parameters to variable arrays to change them without the need to access directly to the model, this makes it possible to calibrate the models easily and fast.

Models

After this theory introduction, now it is time for an in-depth view of the models. Starting from the overview of the progressive version of the plant, then proceeding with a detailed description of the singular component's model

From a very general view the plant can be described by the picture below (Fig. 38). In this overview it can be visualized the closed loop, starting from the user input the "Target Speed", the value inserted is compared with the feedback coming from the motor, and depending on the cases the controller decides to increase or decrease the reference currents, comparing them with the stator currents coming again from the motor's feedback, and giving as output the driving voltage for the motor. Using Clarke & Park inverse transform and the rotor position, the voltage is transformed from the V_{dq} rotating frame to the V_{abc} static frame. These voltages are then used to generate Three PWM signals to drive the inverter. The inverter's source is an ideal battery that has not been simulated and is represented by a constant value of the input voltage. Finally, the three-phase ac output of the inverter goes to the squirrel cage induction motor model that converts them into stator and rotor current and mechanical movement. To close the loop, feedbacks are taken from the motor and given to the controller.

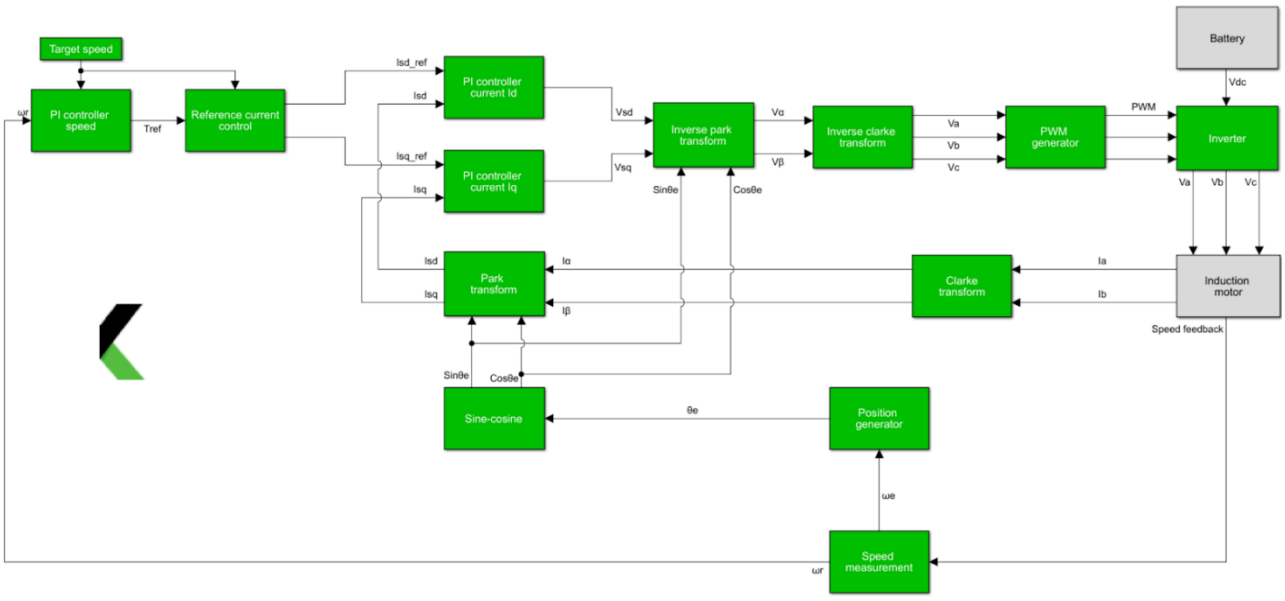


Fig. 38 General view of the Plant

Progressive Plants

To achieve the goal of the thesis or the final plant, we organize the work in plants with progressive complexity. This decision was taken to help us in the debug in case of errors or malfunctions, which can be easily found in simpler models. This approach was also useful for us to practice with the new software and hardware used for this project.

HIL Version V1_1

The first version is a very simple system implementing a controller on the FPGA, and the electrical and mechanical model of the SCIM on the processor. In this version, all the voltage and current are already in the d-q frame, so there is no need for transformation.

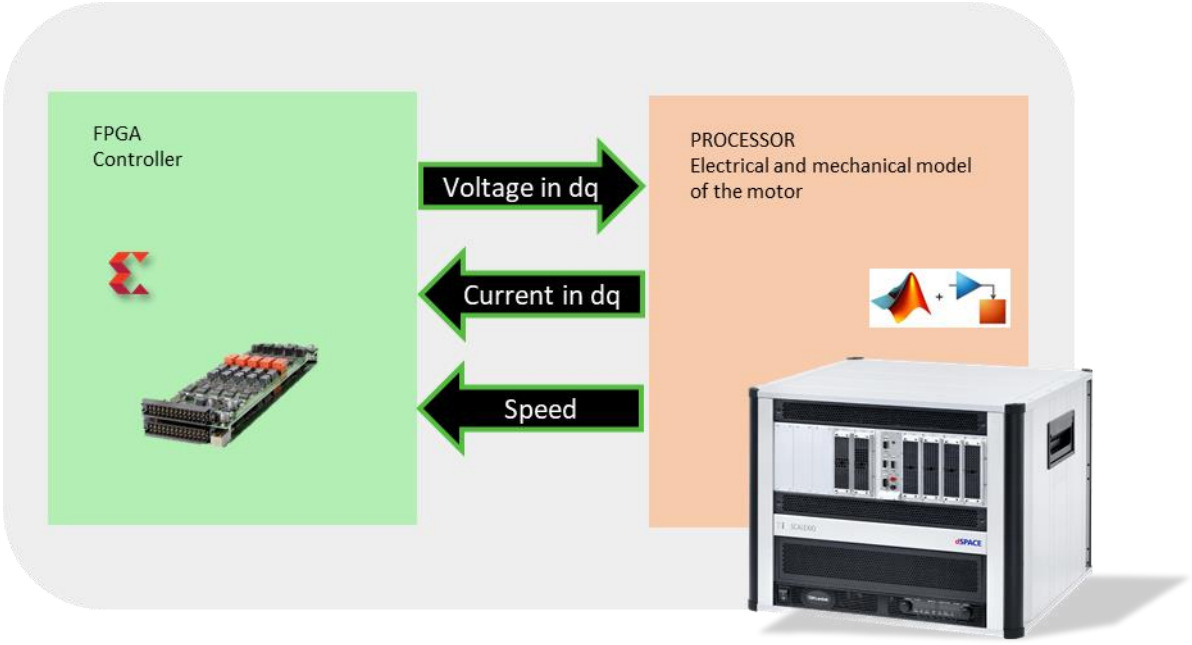


Fig. 39 HIL Version V1_1

The communication between the two hardware is handled internally so there is no communication with the extern. The signal exchanged are expressed by the green/black arrows.

HIL Version V1_2

The second version is more “HIL oriented” where there is a first trial to communicate with an external device and handle real voltages and currents through a wiring harness. It is an upgraded of the version 1_1 with an APU to calculate the rotor position and find the angle θ and the Clarke & Park transformation to calculate the V_{abc} that are then sent out through three analogical outputs and with simple bridges reinserted as analogical inputs. The output is also plotted on an Oscilloscope.

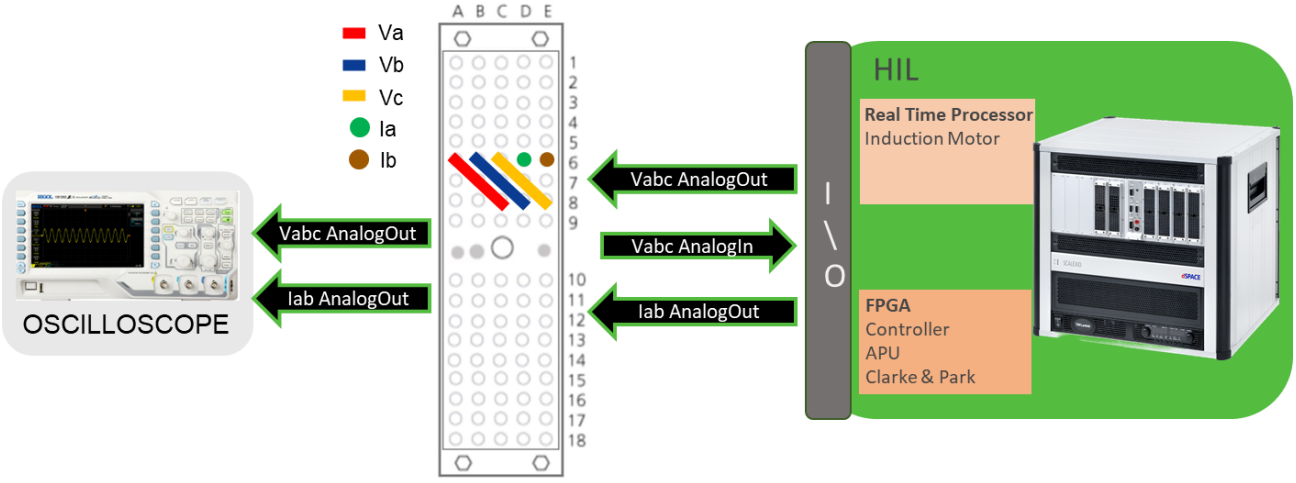


Fig. 40 HIL Version V1_2

Formally this version does not add any new complexity to the system, it is more of an exercise for the handling of real electrical signals.

HIL Version V1_3

The third version implements a new component such as the inverter and the PWM generator. This is the last configuration that we were able to test since the controller is still simulated. In this model, the PWMs are generated in the FPGA and sent out with digital outputs to simulate an external device and resent in through simple bridges with digital inputs. Then with the PWMs and the V_{dc} of the battery the inverter generates the three-phase V_{abc} for the motor.

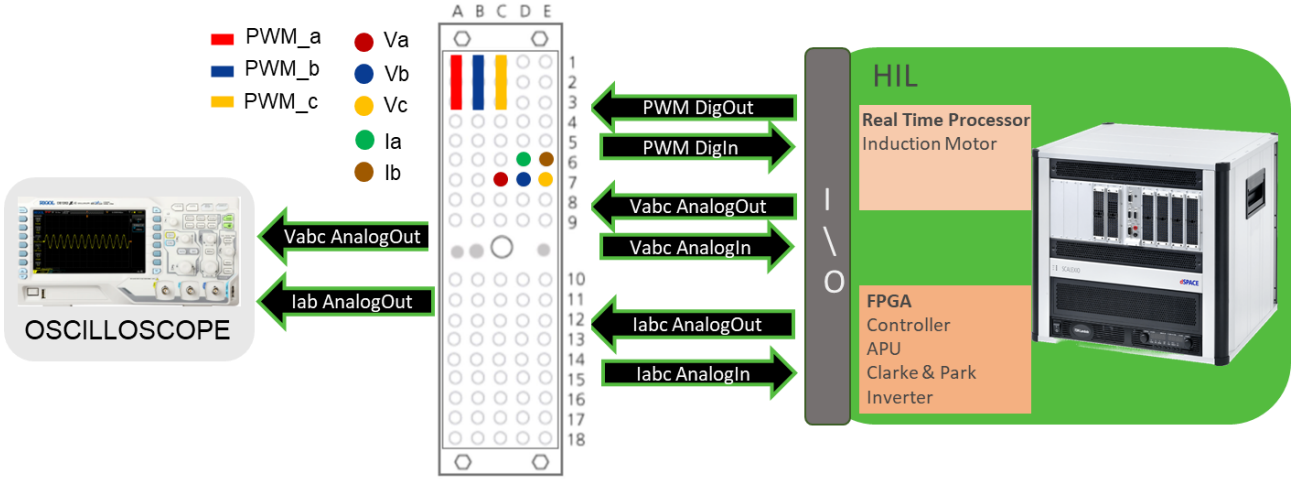


Fig. 41 HIL Version V1_3

This passage of taking out the PWMs and resent them back without doing anything may seem useless, but this is because the model is already "HIL configurated" which is also the thesis's goal and the fourth and last version of the plant.

HIL Version V1_4

The fourth and last version is pretty much the same as the V1_3, but without the simulated controller. The system is configured to accept 3 PWMs and gives as feedback two currents I_{ab}, so the system can be connected to an external device like an ECU and act as an induction motor.

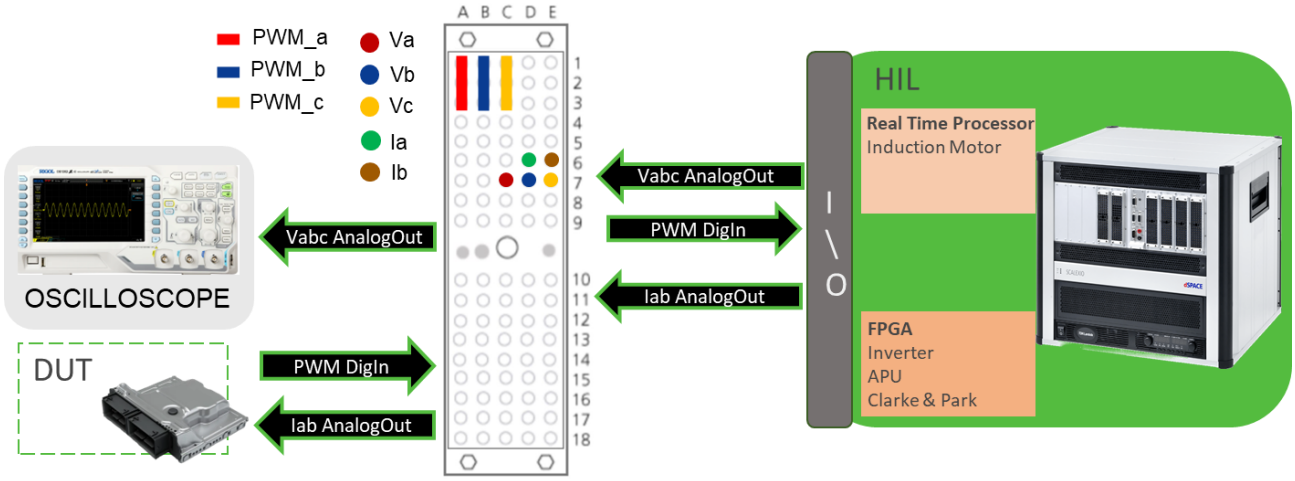


Fig. 42 HIL Version V1_4

With this plant is possible to perform tests on real controllers by monitoring the engine performance and its response to the ECU control. This made it possible to understand the correctness of the control software as well as calibrate it to improve performance and accuracy.

Although this plant is the goal of the thesis, unfortunately, we were not able to test this configuration since the absence of a real device to put under test.

Now that a general overview of the plant has been done, let us see a detailed description of the various component's models that makes the system work. In this section, we will analyse the models, the mathematical function behind them, and the modelling choices.

Clarke & Park Transform

As previously said the control algorithm chosen to drive the motor, is the Field Oriented Control (FOC), which means controlling the motor in the rotating d-q frame. To transform from the stationary three-phase abc frame to a rotating d-q frame the Clarke and Park transforms are used.

The figure below is a snapshot directly taken from the plant, in this case, is used to convert a tension V_{abc} to V_{dq}, but it is also used for the conversion of currents I_{abc} to I_{dq}. The blocks are the same, only the input and output change.

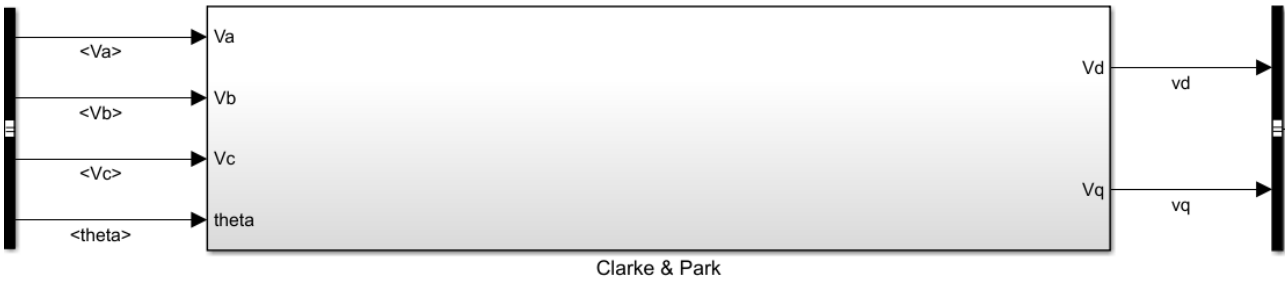


Fig. 43 Clarke & Park Block

Inside there are two subsystem which represent the two separate transform.

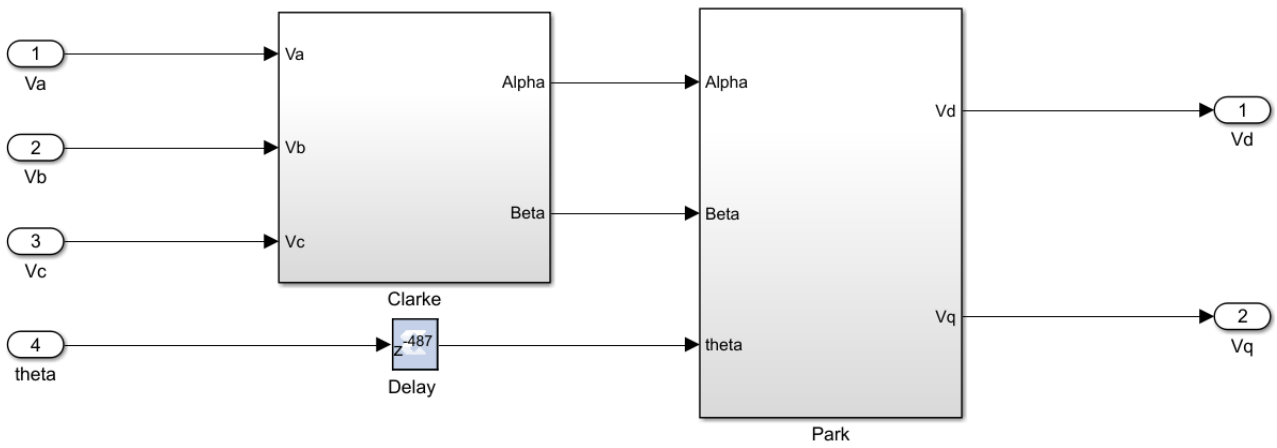


Fig. 44 Clarke & Park Subsystem

As can be noticed there is a delay block with an unusually high value, this is because the FPGA is not ideal and cannot perform instantaneous computation. The blocks are at the end of the computation,

and while the angle theta is obtained from the rotor speed through the APU and requires less computational delay, the V_{abc} is generated more passage (controller, PWM generator, inverter). For this reason, the Delays need to be balanced.

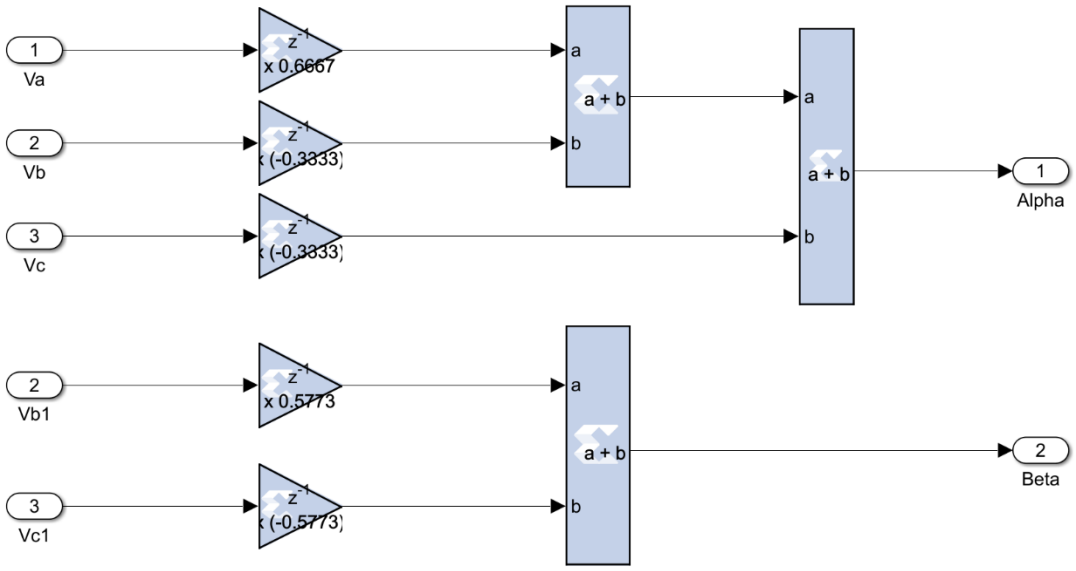


Fig. 45 Clarke

These are simple mathematical passage that can be represent by the equation below:

$$V_{\alpha\beta}(t) = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}\begin{bmatrix} V_a(t) \\ V_b(t) \\ V_c(t) \end{bmatrix}$$

$$V_\alpha = \frac{2}{3}\mathrm{V_a} - \frac{1}{3}\mathrm{V_b} - \frac{1}{3}\mathrm{V_c} \qquad V_\beta = \frac{\sqrt{3}}{3}\mathrm{V_b} - \frac{\sqrt{3}}{3}\mathrm{V_c}$$

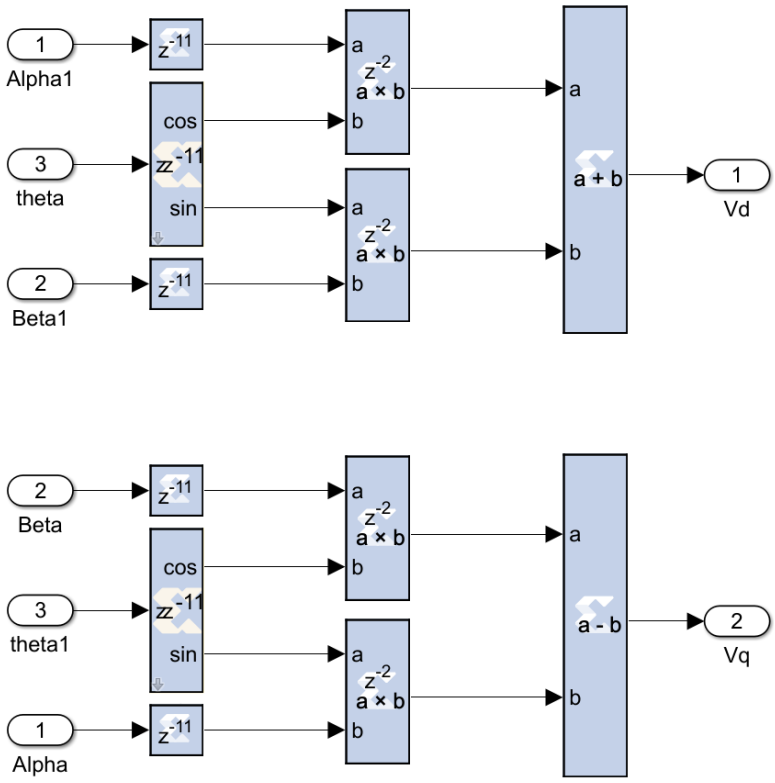


Fig. 46 Park

Again, it can be represented by the equation below:

$$i_{dq} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}$$

$$V_d = V_\alpha \cos\theta + V_\beta \sin\theta \qquad V_q = V_\beta \cos\theta + V_\alpha \sin\theta$$

In the model also the inverse of these transformations is used, from V_{dq} to V_{abc}, while for the park the equations remain very similar but change the input and output, for Clarke, there is a little difference:

$$V_a = V_\alpha$$

$$V_b = \frac{\sqrt{3}}{2} V_\beta - \frac{1}{2} V_\alpha$$

$$V_c = -\frac{\sqrt{3}}{2} V_\beta - \frac{1}{2} V_\alpha$$

APU

From a design point of view, this transformation could have been modelled on the processor and not on the FPGA, if not for the angle θ which can reach higher dynamics than could not be managed by the processor which has a step-size of one millisecond. This choice was taken more in the prevision of future improvement of the motor's performances. For example, at 10000 RPM we have 166 RPS and almost 0.166 rotation every millisecond. So, every 6 milliseconds there is a rotation, which means that the angle change from minus π to plus π, will be divided into six steps, already with very low precision.

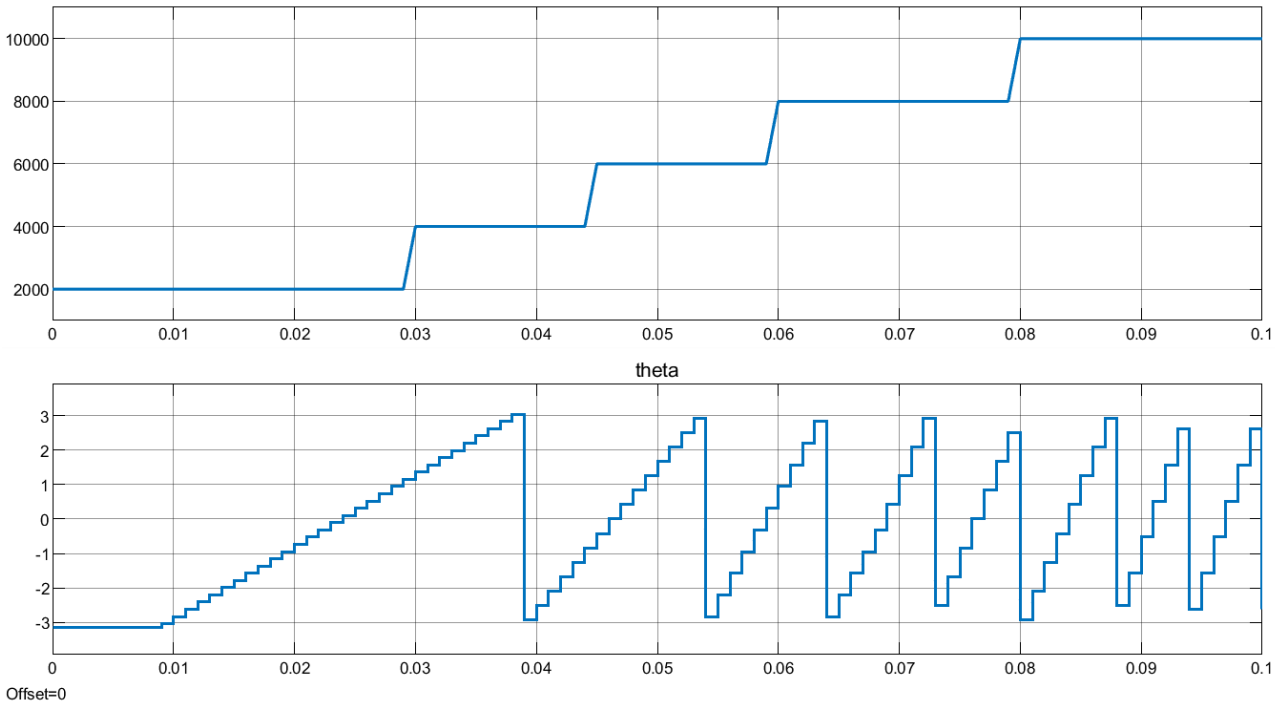


Fig. 47 Theta at 1 ms Step-Size

As can be noticed, increasing the speed corresponds also to a loss of precision, so much that could make the usage of the FPGA useless.

That is the reason why we decided to design the Clarke and Parke transform in FPGA, and that is also the reason why we needed to design another component of the APU.

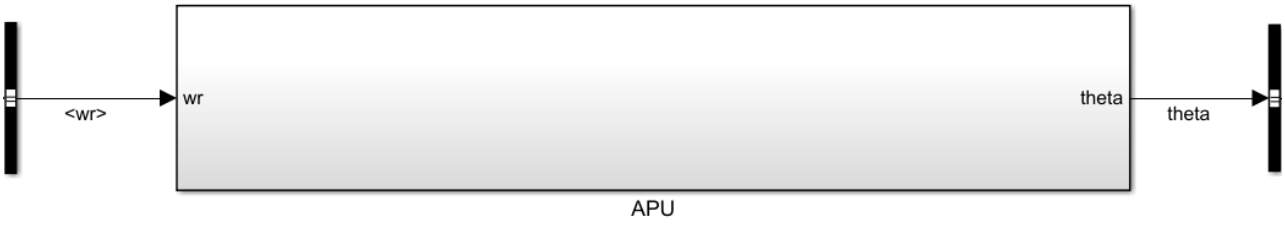


Fig. 48 APU

The APU is a simple component that takes as input the Rotor's speed and computes the angle θ wrapped around $-\pi \; and \; +\pi$ which is given as output.

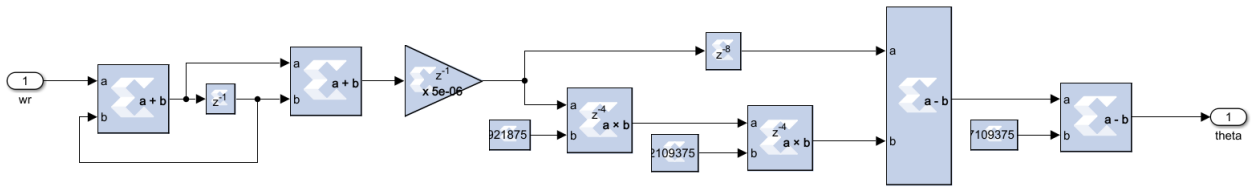


Fig. 49 APU Inside View

Firstly, an integration is performed, and the result is a ramp that needs to be wrapped around minus and plus pi. To do this we transform the ramp into steps of 2π as follows:

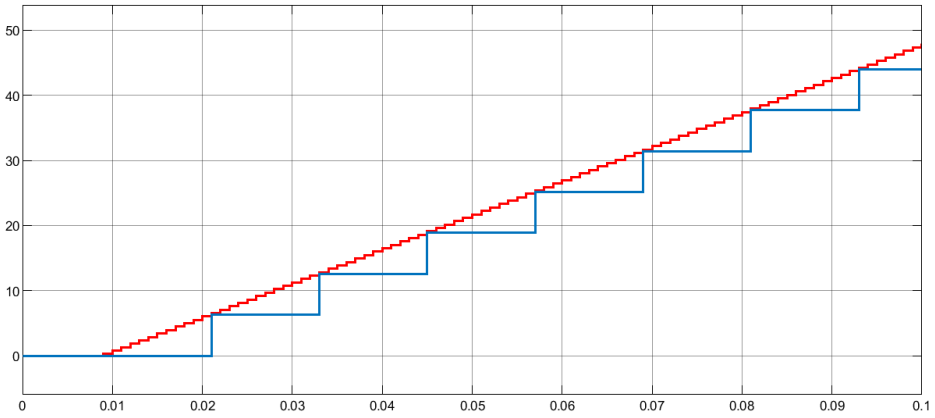


Fig. 50 Ramp division

With these signals, a subtraction is then performed, and it's obvious that doing this will wrap the output from 0 to 2π so another subtraction is preformed to obtain the desired output.

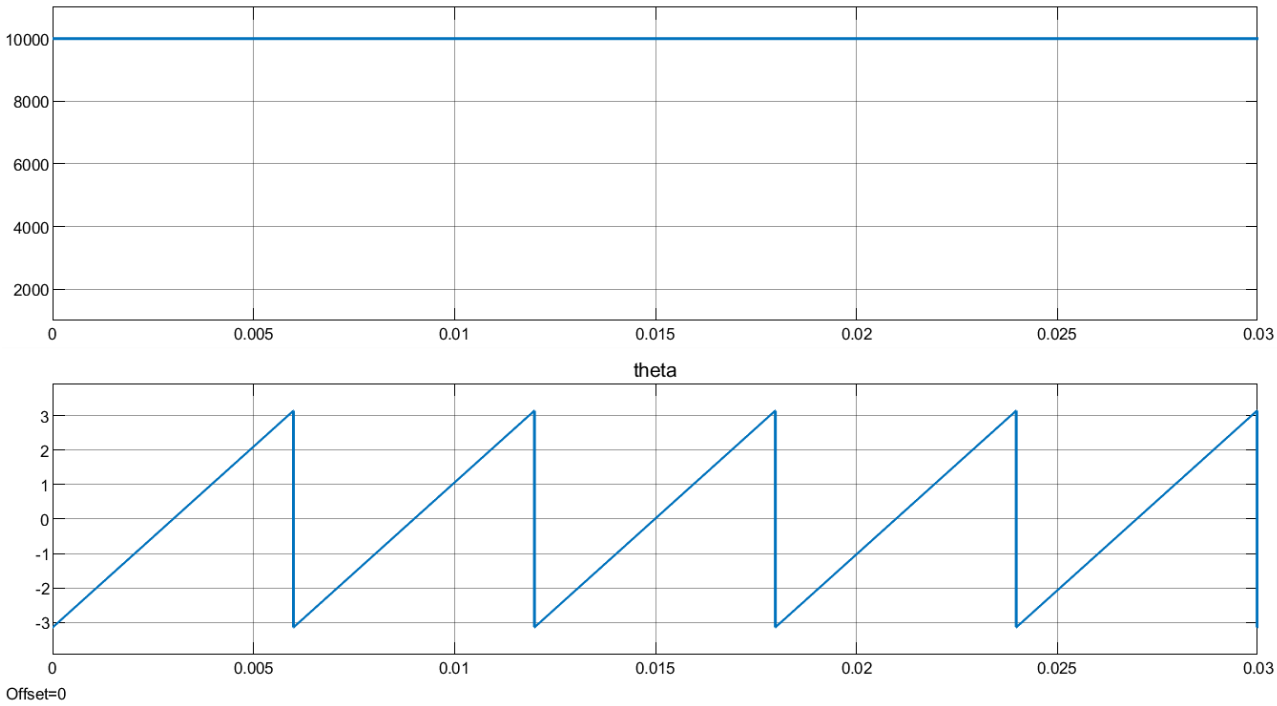


Fig. 51 APU Output

Inverter

The inverter, along with the electrical motor, was the centre of the thesis, and the from a design point of view also the one with more difficulties. As previously mentioned, it is a three phase RC inverter, and due to the high frequencies reached from the PWM it was deployed on the FPGA.

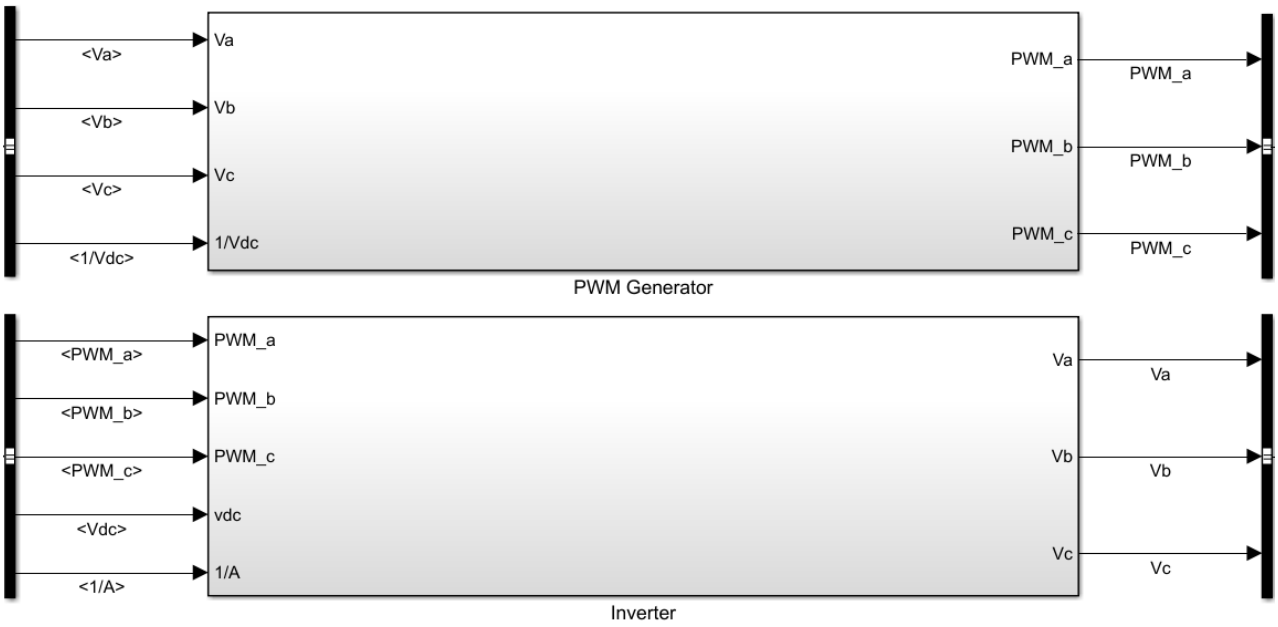


Fig. 52 PWM Generator & Inverter Overview

For the definitive plant (V1_4) the PWM generator is not necessary since the external controller already provide a PWM signal, but without the external device, it is essential for testing the system.

PWM Generator

PWM generation start from the reference three-phase signal V_{abc} coming from the controller, which is multiplied to the inverse of the battery voltage V_{dc}, to be sure that it is never greater than 0.

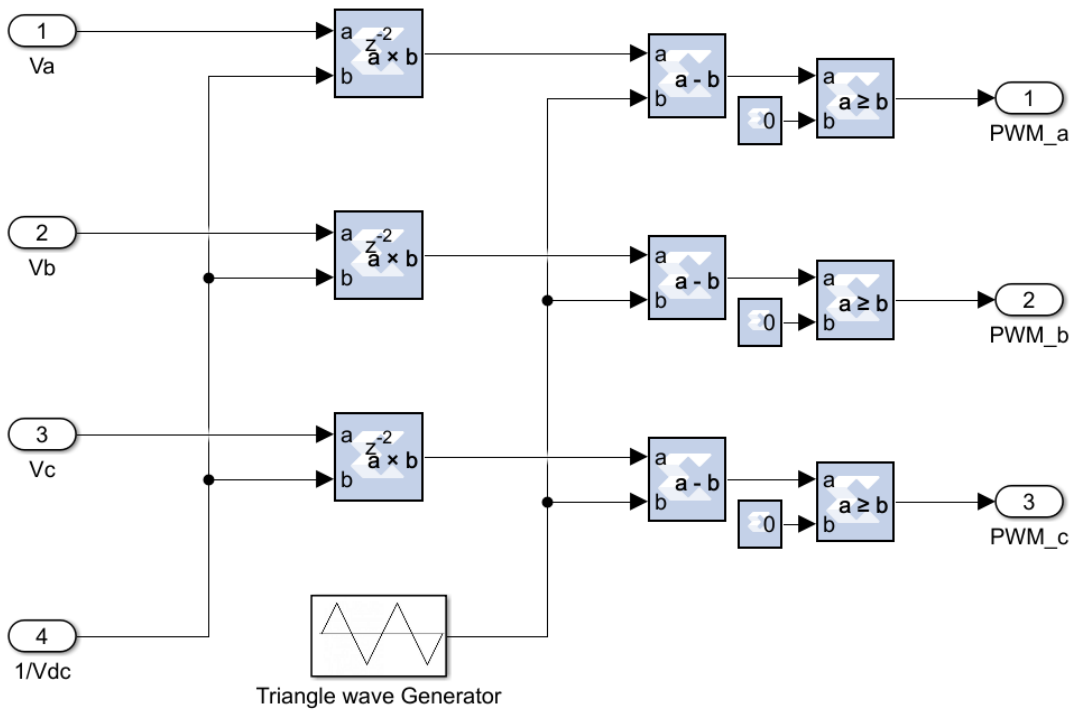


Fig. 53 PWM Generation

Then it is compared to a unitary triangular wave which is generate in another custom block that we had to design in FPGA.

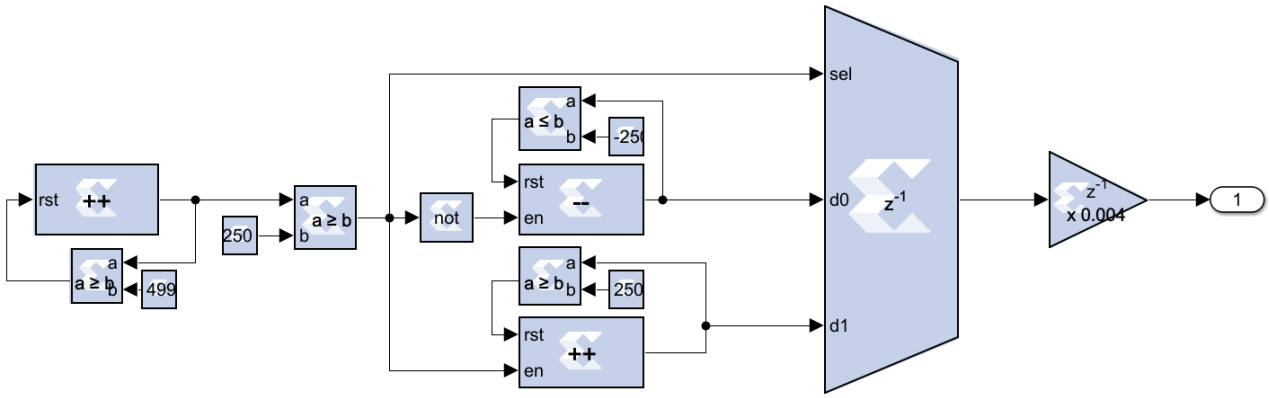


Fig. 54 Triangle Wave Generator

The triangular wave is generated using counters and logical blocks, in few words there are tree counter:

- one counting up generating the “increasing” phase of the wave, and resetting after reaching its peak value

- one counting down generating the “decreasing” phase of the wave, and resetting after reaching its peak value

- one that gives the wave period and that activate and deactivate the other two counter.

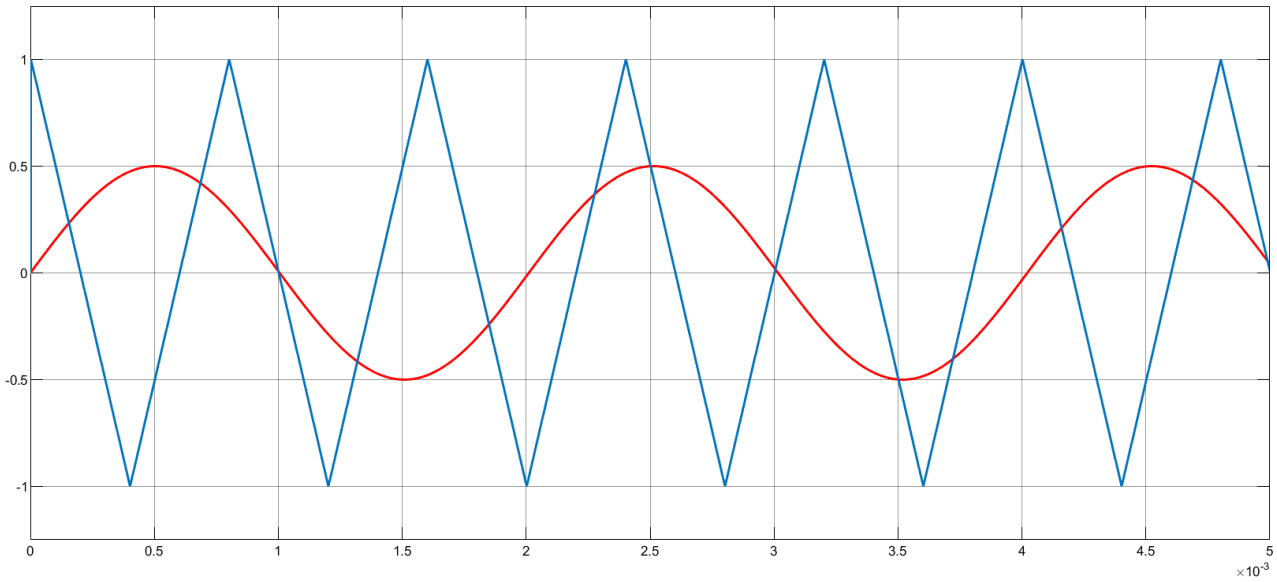


Fig. 55 Triangular Wave

Finally, to obtain the PWM, a comparison to 0 is performed, obtaining:

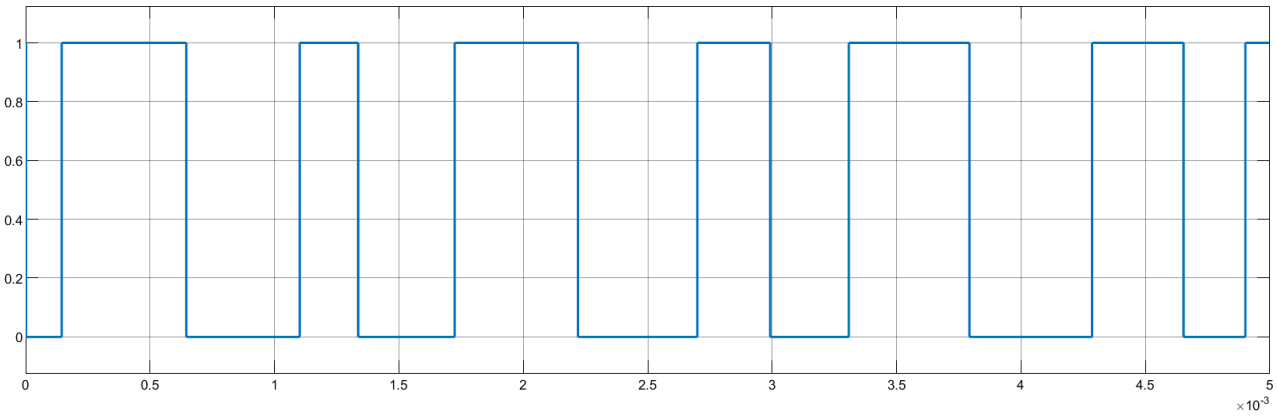


Fig. 56 PWM Signal

With the three PWM generated, now it's time to describe the inverter:

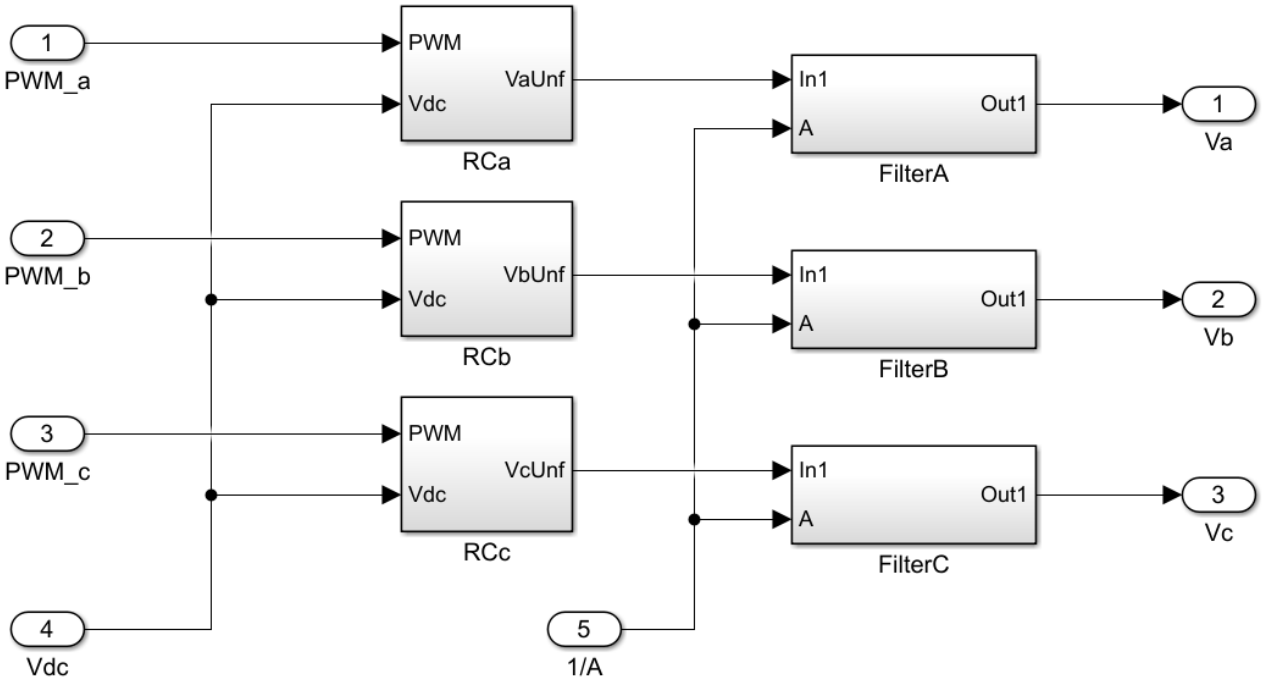


Fig. 57 Inverter

The inverter is composed of two main components: the RC circuit equivalent model, and a filter.

The RC equivalent take as input the PWM which is the control of the output, and V_{dc} or the continuous battery source which also limits the output.

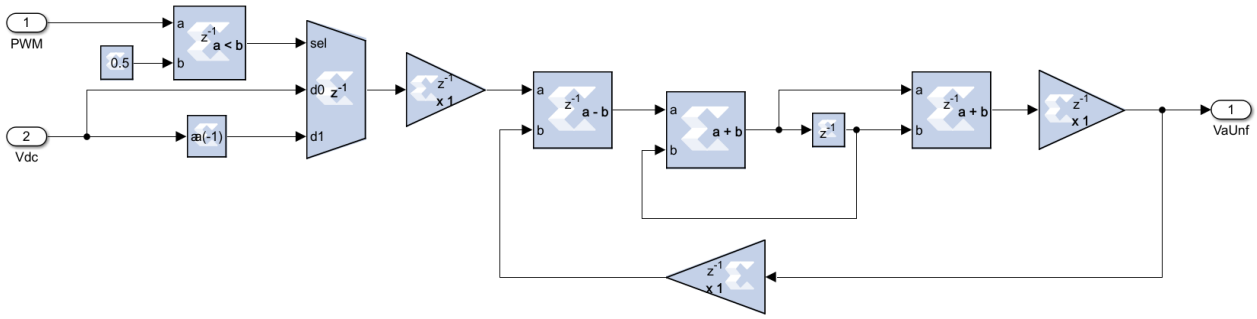


Fig. 58 RC equivalent

The PWM is used to drive a selector that chooses between $+V_{dc}$ and $-V_{dc}$ based on the desired output, positive will result in an increasing sinusoid while negative will result in a decreasing sinusoid. The next is just the equivalent of the RC circuit previously discussed in the theory part, and corresponds to:

$$V_{out} = \frac{1}{RC}\int V_{in}\, dt$$

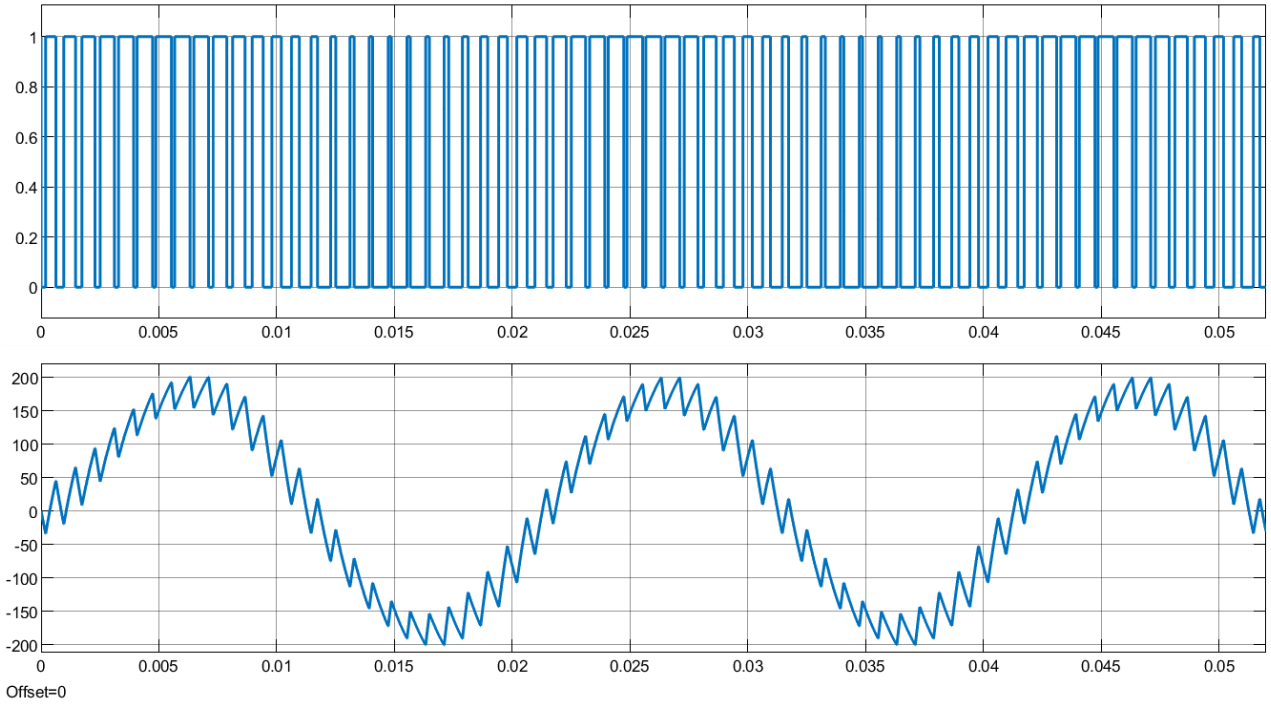


Fig. 59 Inverter Output

The output is the classic behaviour of an RC circuit, so to have a better and more precise result a filter is added. Furthermore, the RC integrator also adds an attenuation which is not constant, but increases proportionally with the frequency and so, the motor speed.

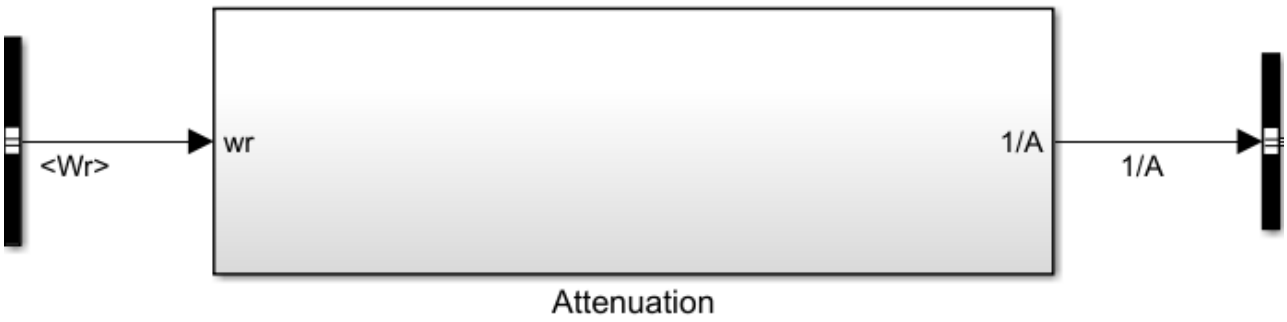


Fig. 60 Attenuation

The attenuation is computed directly on the processor using the rotor speed. The formula for obtaining this attenuation is simple and comes from the theory:

$$A = \frac{RC^2\omega_r^2 + 1}{\sqrt{RC^2\omega_r^2 + 1}}$$

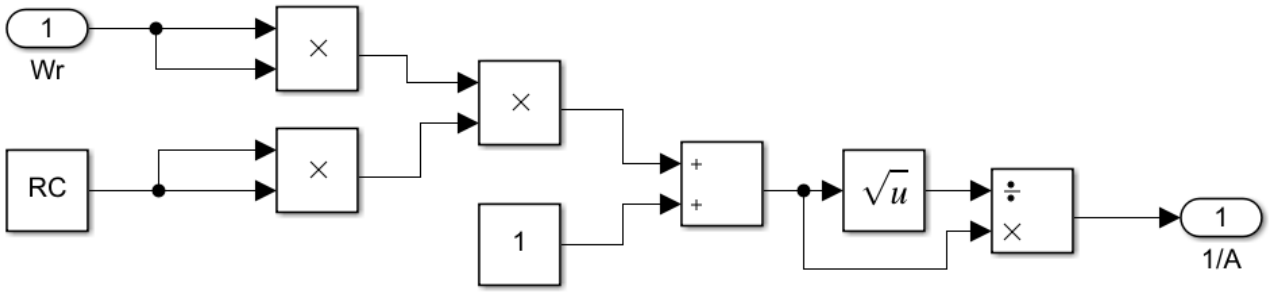


Fig. 61 Attenuation Computation

This value is then sent to the FPGA and used by the filter to mitigate the attenuation generated by the inverter.

The filter has two inputs, the signal to be filtered and the attenuation. Knowing the nature of the signal is generated by the charging and discharging of a capacitor, which is almost similar to a triangle wave, the desired signal can be obtained by computing the mean value.

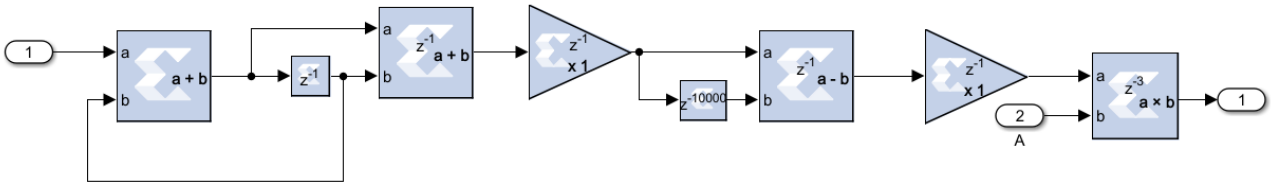


Fig. 62 Filter

So, the filter act as a mean with relation:

$$\frac{1}{T}\int_{(t-T)}^{t} f(t)\ dt$$

Where:

- $f(t)$ is the input signal
- T is the fundamental period or 1/fundamental frequency

The last operation is the multiplication with the reciprocal of the attenuation, the result is shown in the next figure, in blue is the unfiltered signal, and in red is the output result.

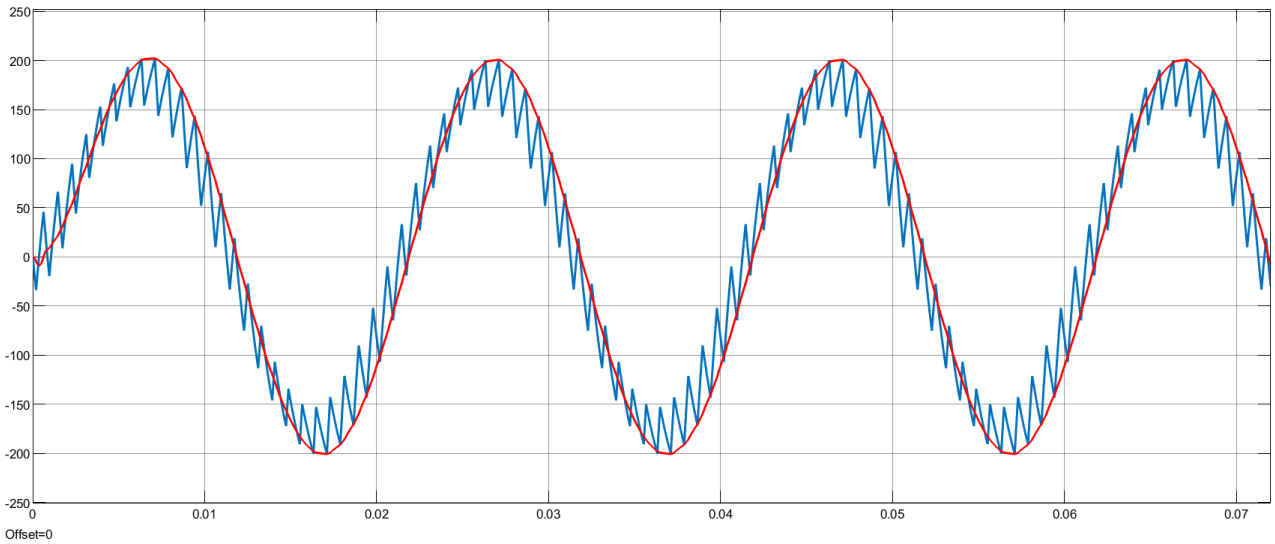


Fig. 63 Filter Output

The result of the inverter is very satisfactory, after numerous problems, difficulties, and tests, we were able to obtain an almost perfect output that reflects with very high precision the behaviour needed for a three-phase ac.

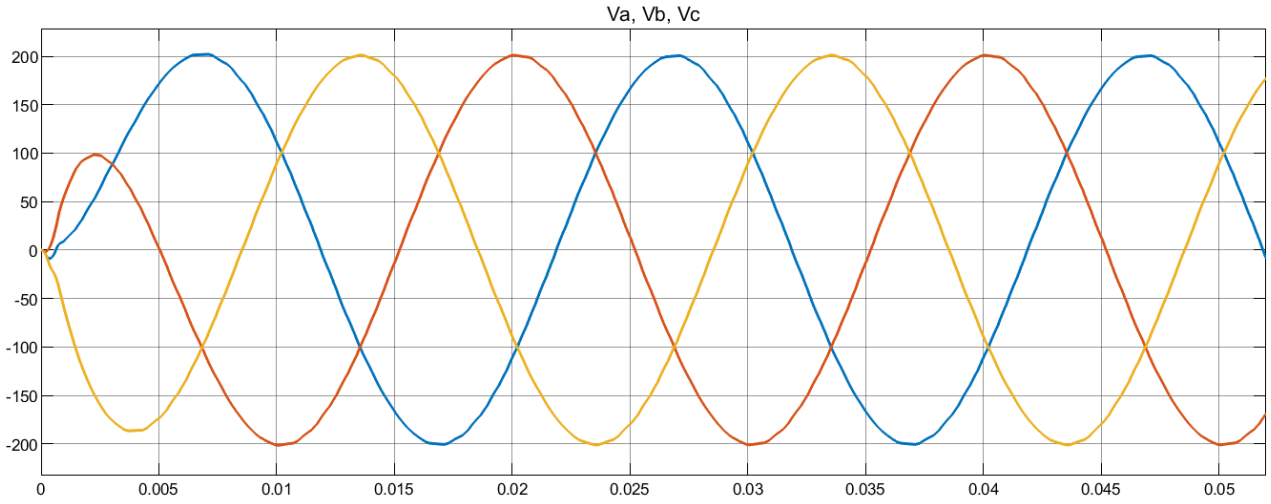


Fig. 64 Inverter Output

Some non-idealises of the RC inverter can be noticed at the start, where the capacitor needs some time to charge and follow the reference signal.

Induction Motor

The three-phase ac generated by the inverter is then transformed in the d-q frame to be sent to the processor where the model of the induction motor is located. This component has been extensively described in the thesis of *Fabio Oreiller*, therefore, in this case, for any further clarification please refer to the reading of his work.

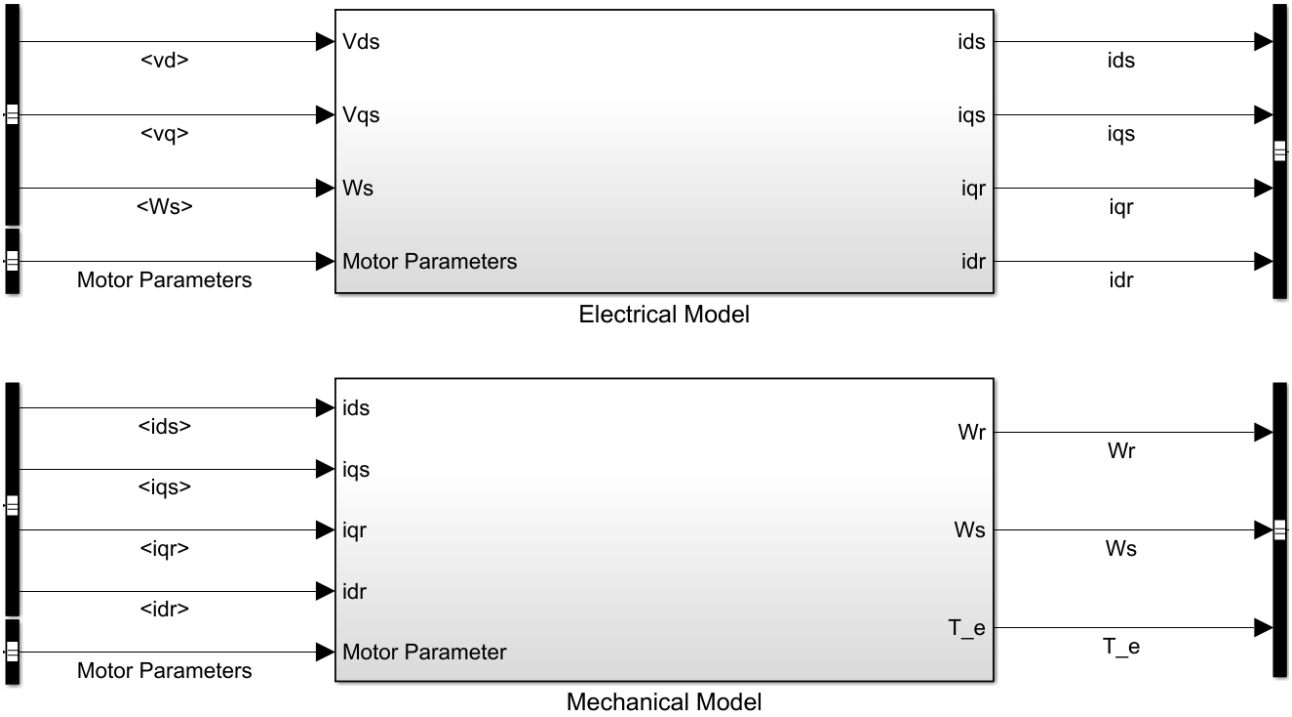


Fig. 65 Induction Motor

The model is divided into two subsystems, one represents the electrical model, and one represents the mechanical model. The models together represent the Induction motor, and as it can be noticed other than the typical input there are the “Motor Parameters”. The presence of these parameters

that can be changed is very important because this system does not represent a single specific motor only but can be adjusted to represent any kind of motor, only by adjusting those parameters.

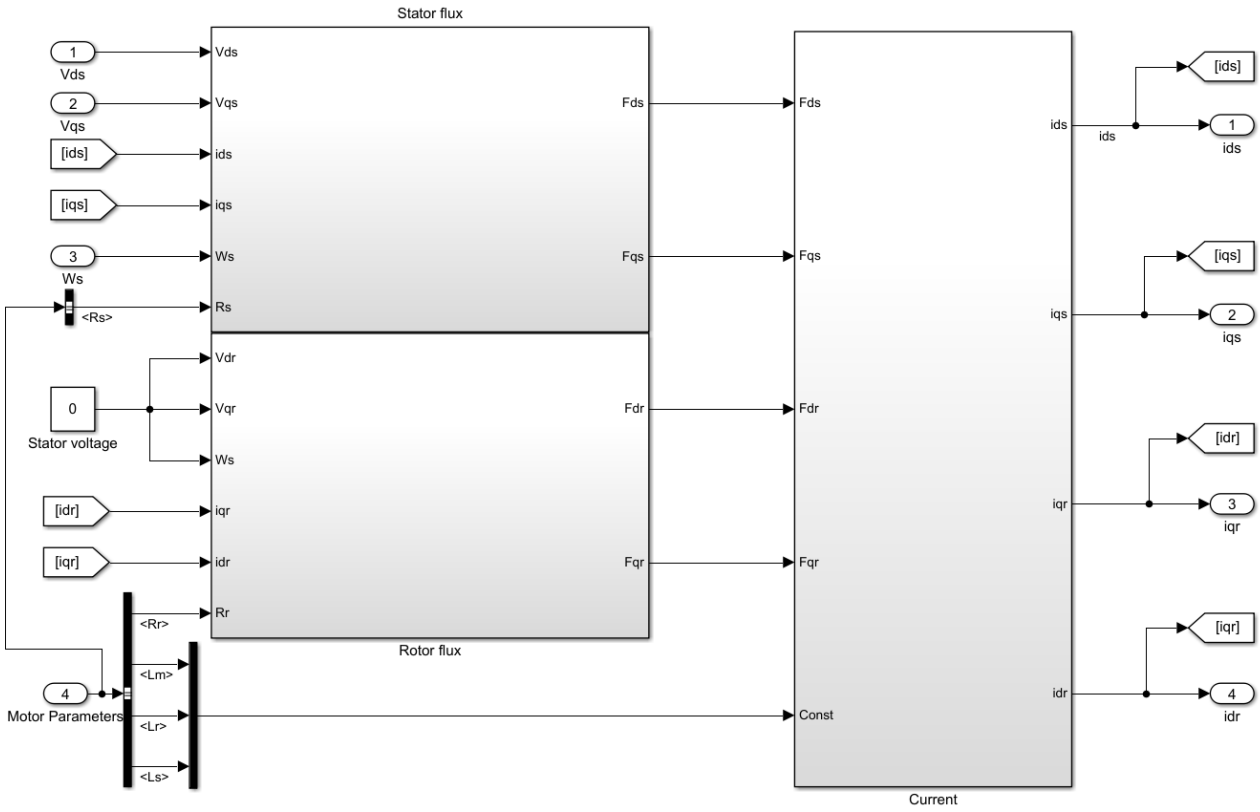


Fig. 66 SCIM Electrical Model

The electrical model takes into input three variable, the two components of the voltage in d-q V_d and V_q and the electrical speed of the stator ω_s, the rotor's voltage are not needed because the windings are short-circuited. After some calculation (for a detailed view and explanation of the equation see the thesis of my colleague) the rotor and stator flux F_{dqr} and F_{dqs} are obtained, proceeding with the operation, finally the stator and rotor current i_{dqs} and i_{dqr} are also obtained.

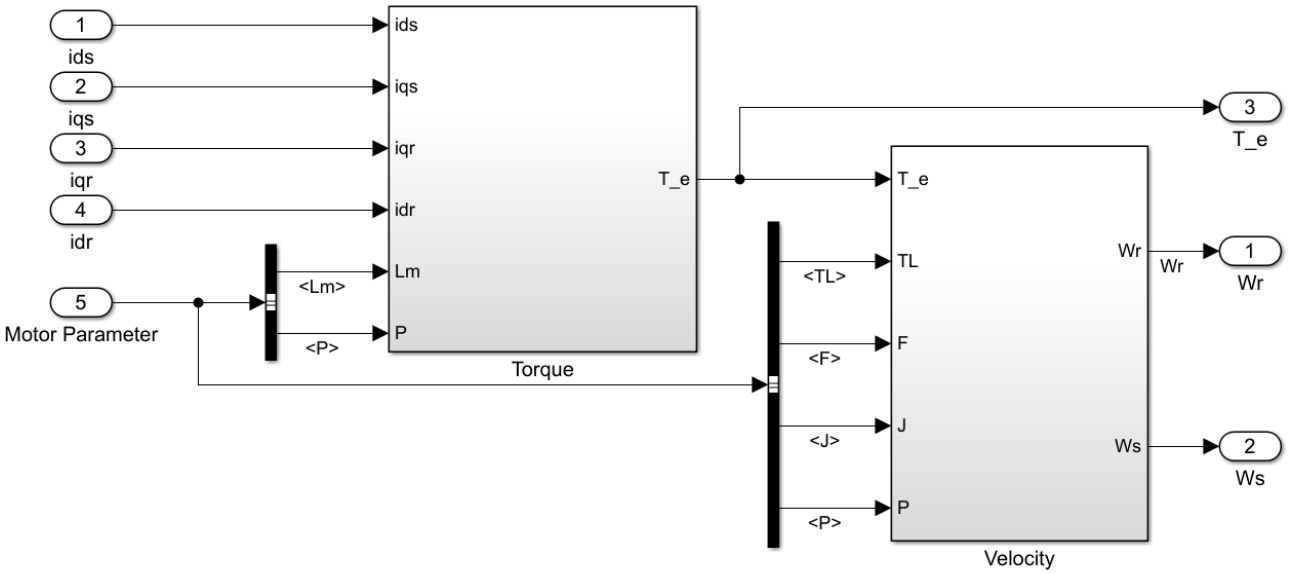


Fig. 67 SCIM Mechanical Model

From the electrical model are obtained the currents necessary for the Mechanical model. From these variables, the electrical torque is computed, which is compared to the load torque to determine the actual speed of the rotor and the stator.

Controller

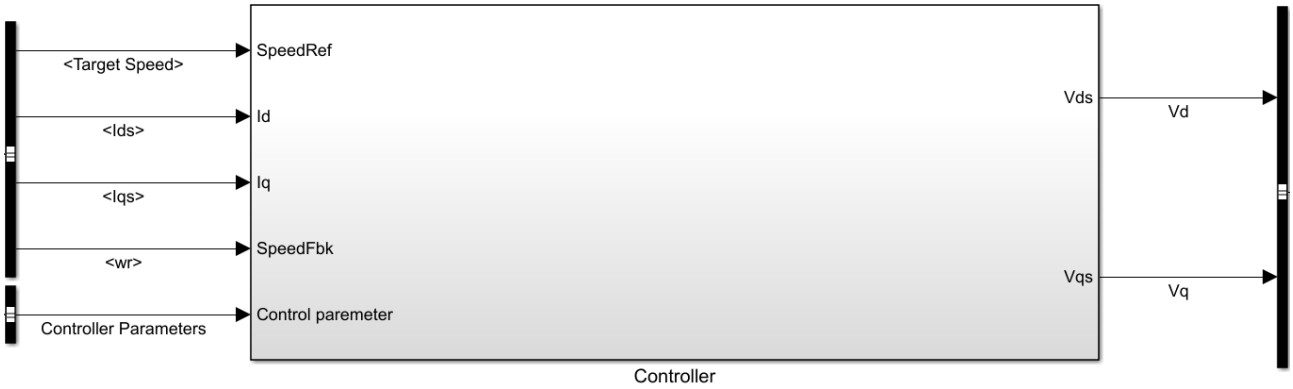


Fig. 68 Controller

Even though the controller was not part of our thesis goal, we still needed to model it because of the absence of a real external ECU. Again, this component has been extensively described in the thesis of *Fabio Oreiller*, therefore, in this case, for any further clarification please refer to the reading of his work. The controller is used to control the behaviour of the inverter and the motor, using a reference speed decided by the user, and the feedbacks of the motor.

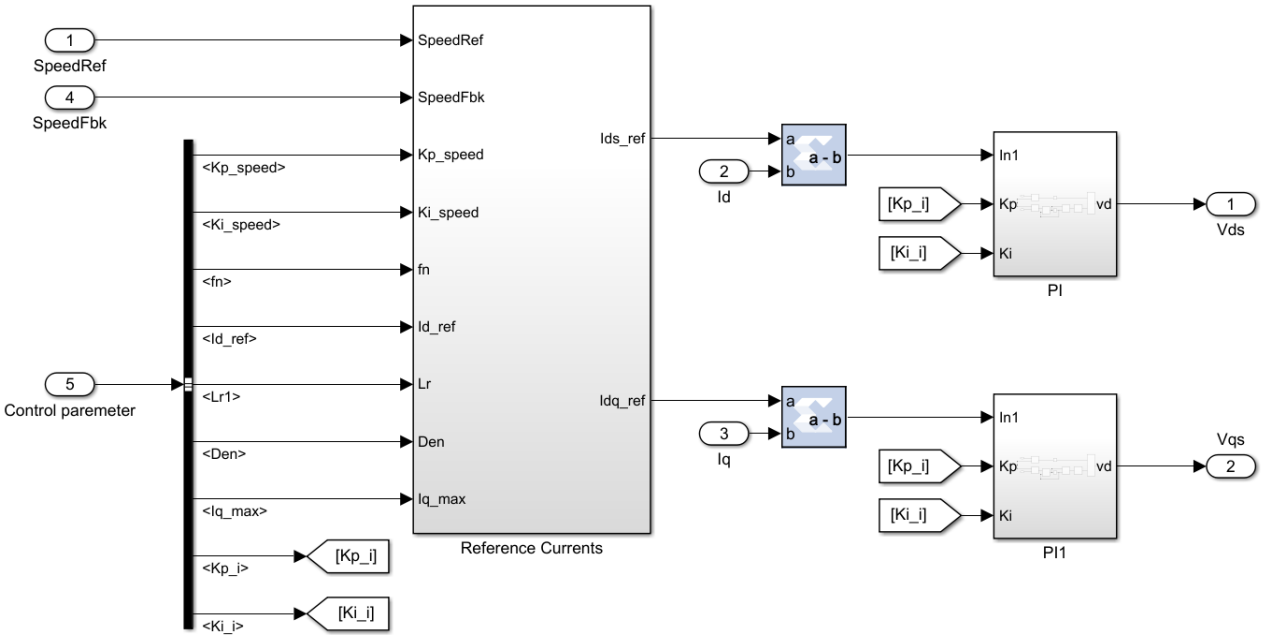


Fig. 69 Controller inside view

It takes has 5 inputs:

- Speed Reference, coming from the user
- Speed feedback, feedback from the motor
- I_{dq}, feedback currents from the motor's stator
- Control Parameters, that decide the behaviour of the control

The controller first compares the two speed and depending on the gap between the two, decide through a PI if increase or decrease the torque. With the reference torque generated, through some computations are also generated the reference current in d-q reference frame. These currents are then compared with the feedback current and through other two PIs the reference voltage that will drive the motor is generated.

Plant Overview

Now that a detailed description of each component has been made, let's view the plant, how it works, how it communicates, and its architecture. The scheme below represents a simplified view of the total plan of our last model, still with the internal controller.

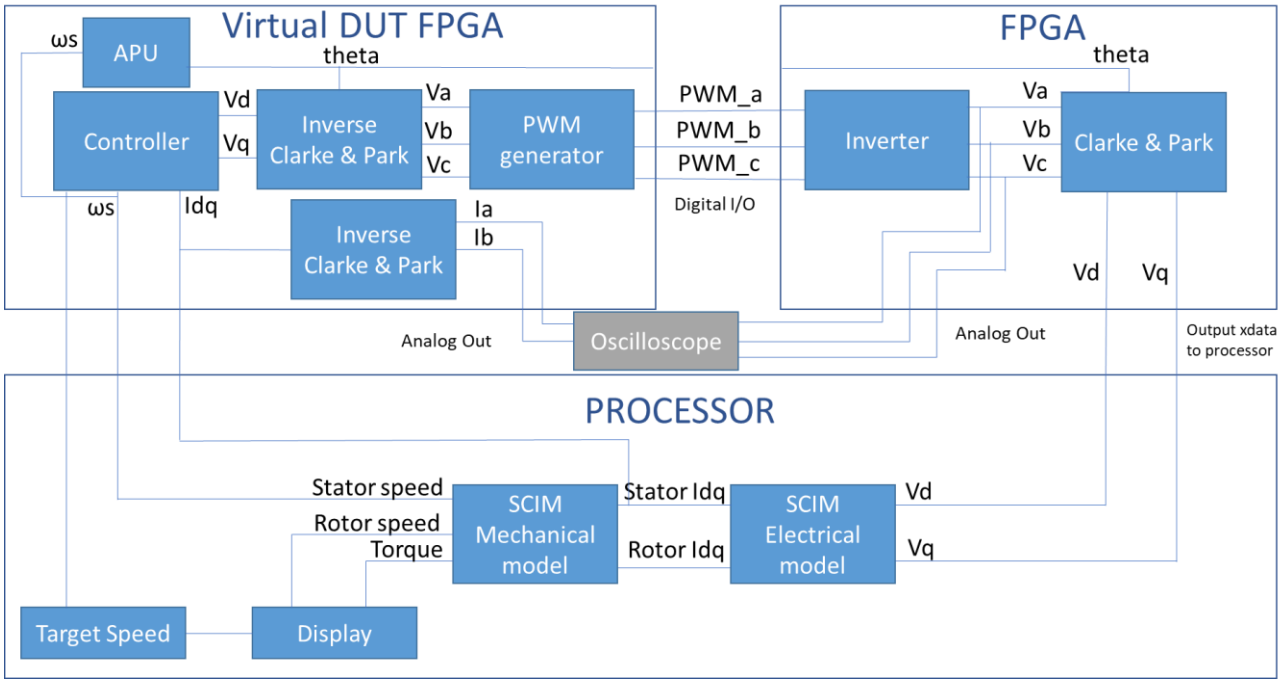


Fig. 70 Plant V1_4 Overview

This is a detailed scheme that want to give a precise idea of the interconnection between every single block as well as the communication between the processor, FPGA, and oscilloscope. The decision to divide the virtual DUT from the rest of the component on the FPGA, comes from the last version of the plant.

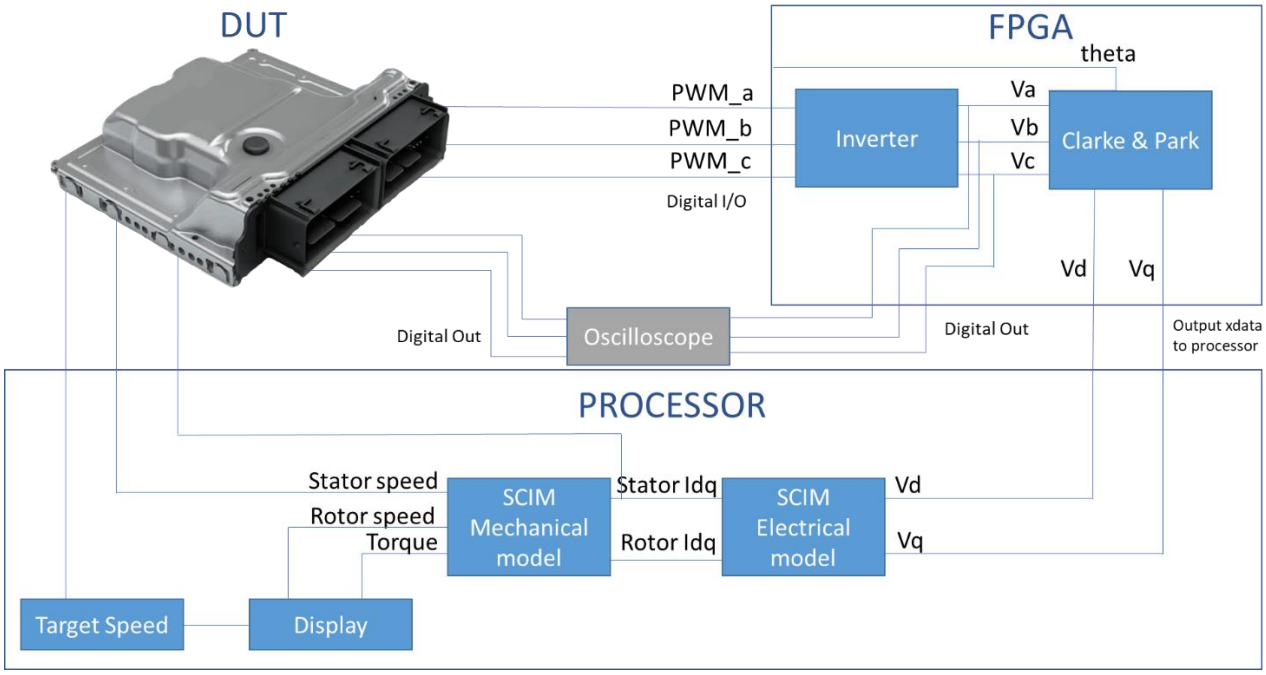


Fig. 71 Plant V1_5 Overview

As we can see in this version a real DUT take the place of the virtual DUT, while all the other components remain the same. This process takes place in a very easy way because the previous version was already configured for this change.

Simulink Design

Let's now take a look at the Simulink design for the plants, we decide to standardize a model to every version, to simplify the debugging and visualization. This is the highest level where we find two first subsystem that divide the models for the hardware which they will be deployed on, the processor, and the FPGA.

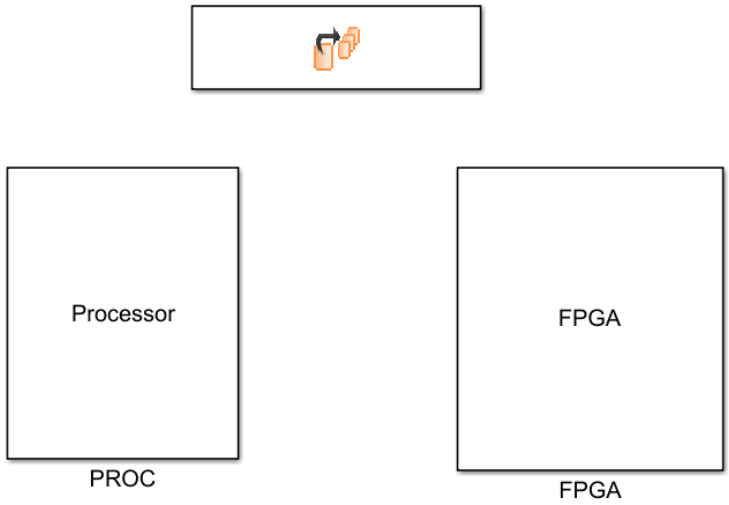


Fig. 72 Plant division

By entering in one of the blocks we will be find a similar situation with many other subsystems. As it can be noticed from a higher level we will go through different block to arrive at the singular block, this division was made to help us have a clear view of the models while working on it.

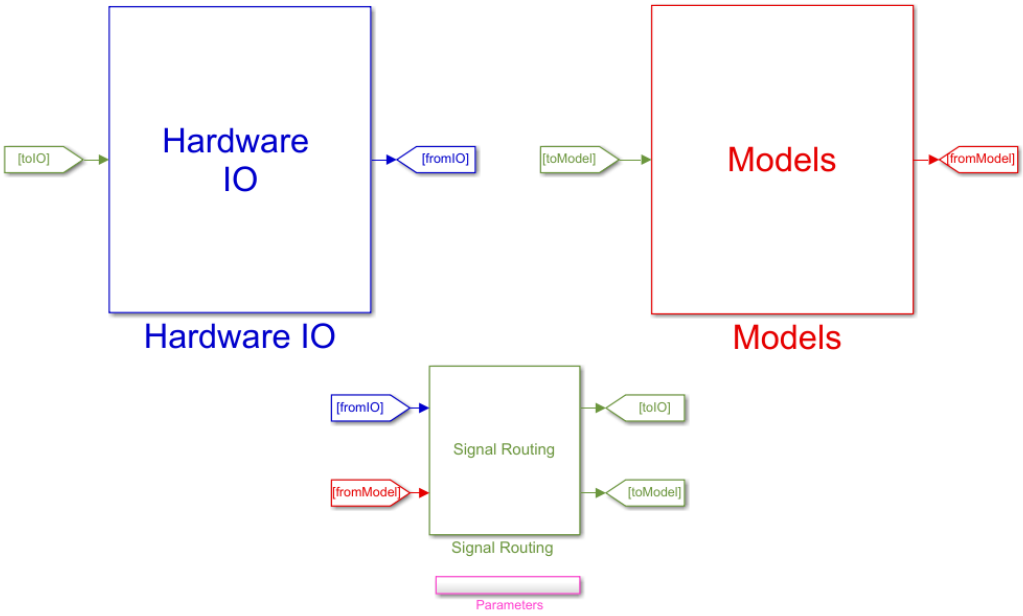


Fig. 73 Standard model

In detail every subsystem has a specific function:

- Models: this subsystem is the core of the plant, it contains all the models

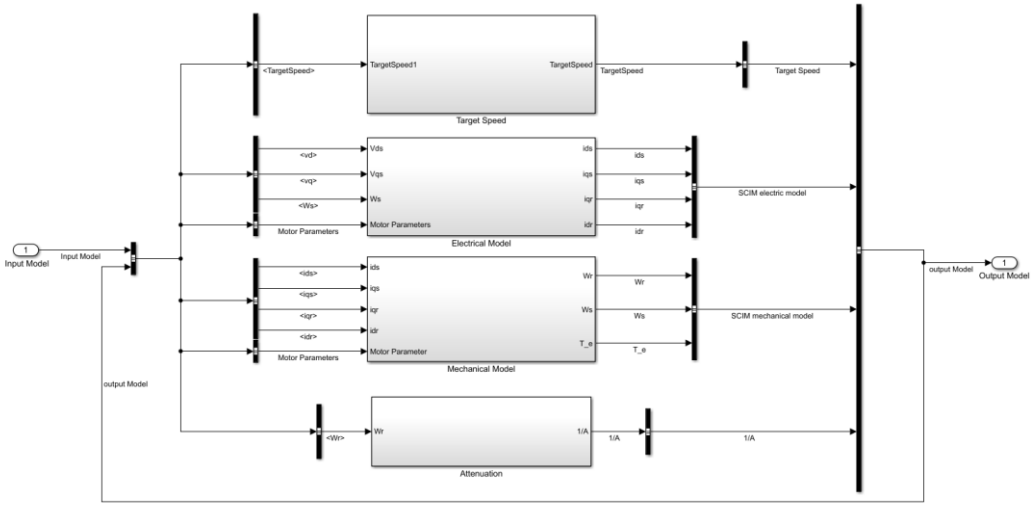


Fig. 74 Models

- Hardware IO: in this subsystem is defined the communication between the processor, the FPGA, and the extern. In detail it specifies all the input and output linking it to the right quantities from the model and the right pin of the board.

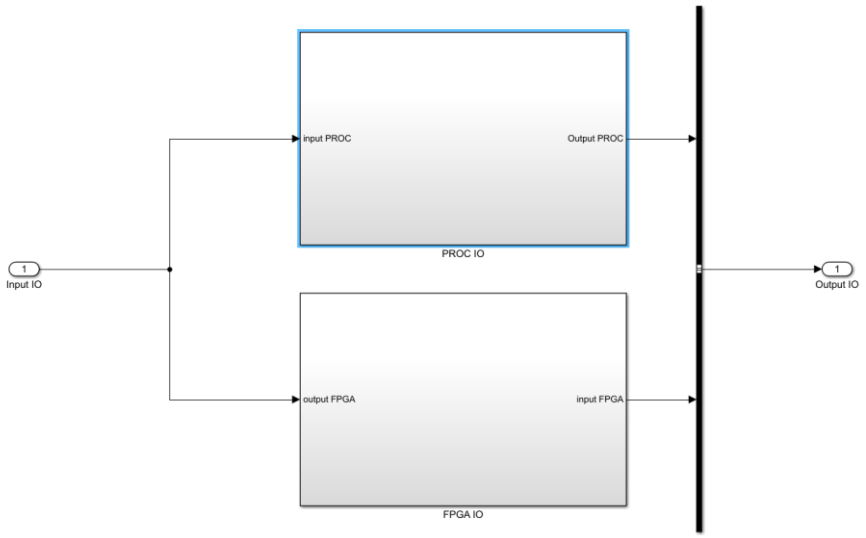


Fig. 75 Hardware IO

- Signal routing: this block is to route all the variable utilised in the other subsystems. The block “Display” is for visualize the information that are needed to be monitored.

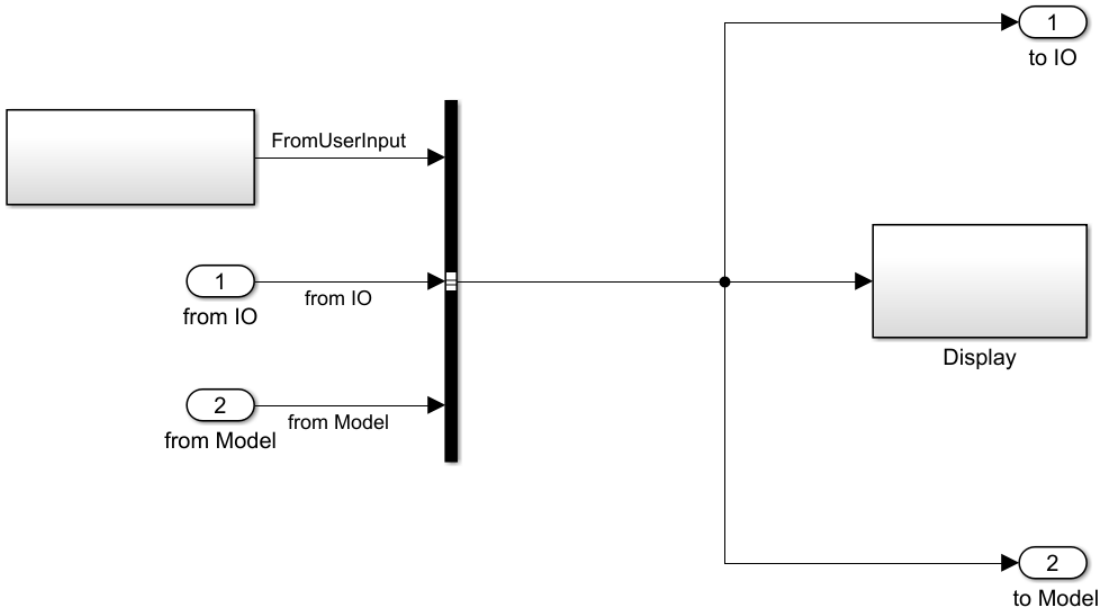


Fig. 76 Signal routing

- Parameters: store all the constant variable that can be modified by the user for changing the behaviour of the control, or of the motor.

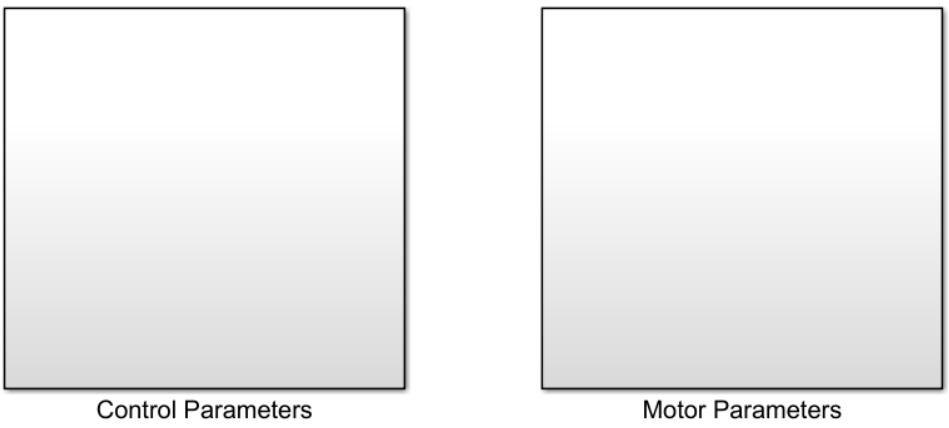


Fig. 77 Parameters

Software Procedure

Now that the model has been widely described, let see the procedure to prepare all the components for the FPGA and the implementation for the SCALEXIO simulator. The first step was the offline simulation on Simulink of the various function, making use also of the Xilinx blockset for the FPGA programming.

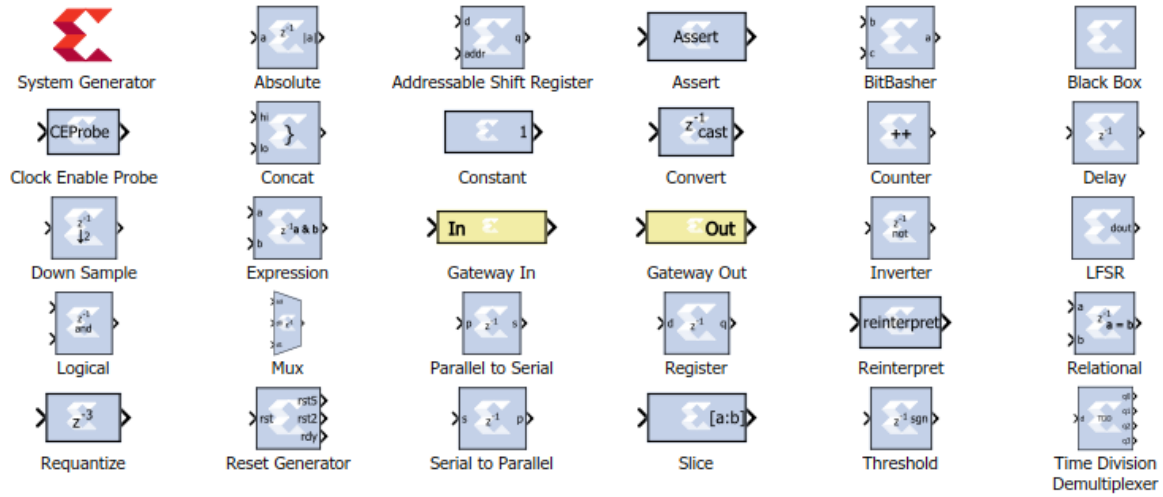


Fig. 78 Xilinx Blockset

The second step was to prepare the plant for the implementation in the simulator, using the dSPACE RTI blockset. Those blocks were needed to specify the type of hardware present in the simulator and each physical pin of the board used for the communication.

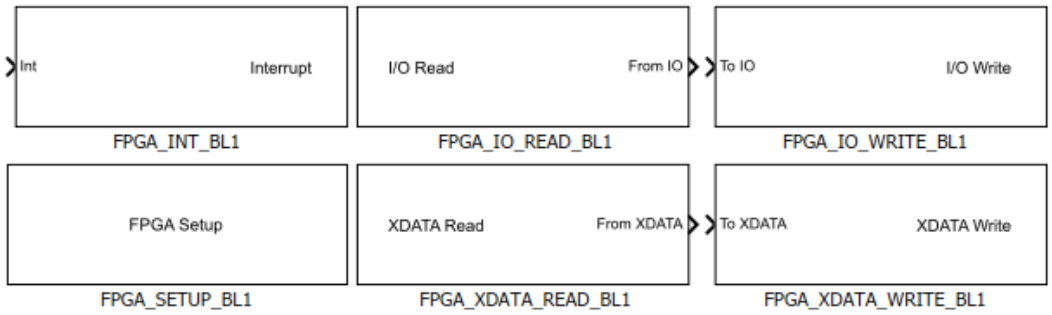


Fig. 79 dSPACE RTI blockset

The result is the division in two main subsystems as previously shown in figure 72.

Timing analysis

Next step was to calibrate each block and function to work on an FPGA. As we know the FPGA is a real component that work in real-time, for this each operation takes a while to be performed. That's why we need to allocate enough time to a certain operation to be performed completely, this time is represented by the latency of an operation, which literally means how many clock ticks are needed.

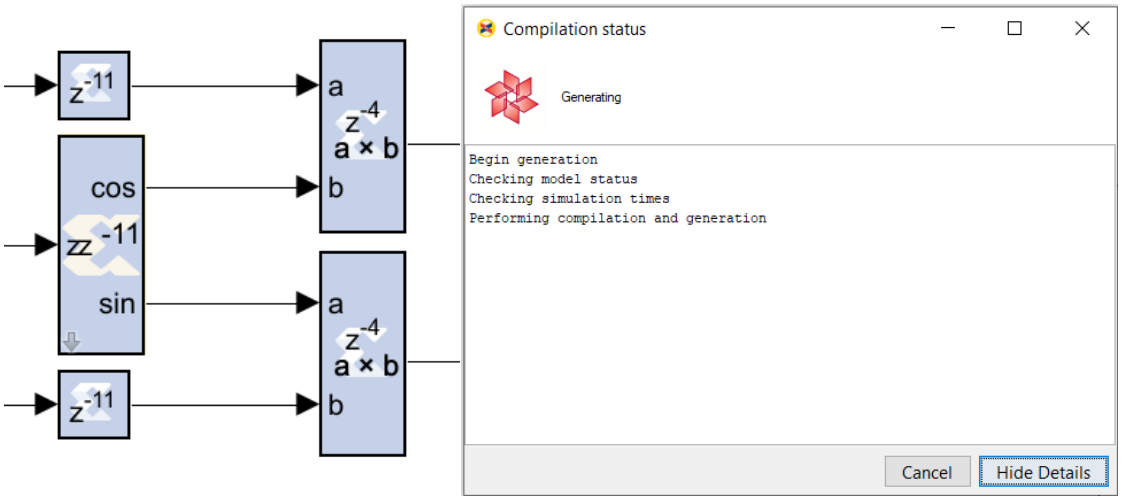


Fig. 80 Example of operation with latency and delay and the analysis process

The figure above shows an example of operation, the block for the sine and cosine will take by default 11 "ticks", this means that after receiving a signal in input, it will give an output after that time. This create a delay in the operation for some signal that must be balanced with the "delay" blocks (in the picture there are one above and one below the sin-cos).

After calibrating all the models, it is necessary to check if the calibration was performed in a right way, in order to be sure of not having errors and problem on the simulator. So, the next step is to perform the "Timing Analysis" which will control each block and generate a report to tell us either that the system passed or not.

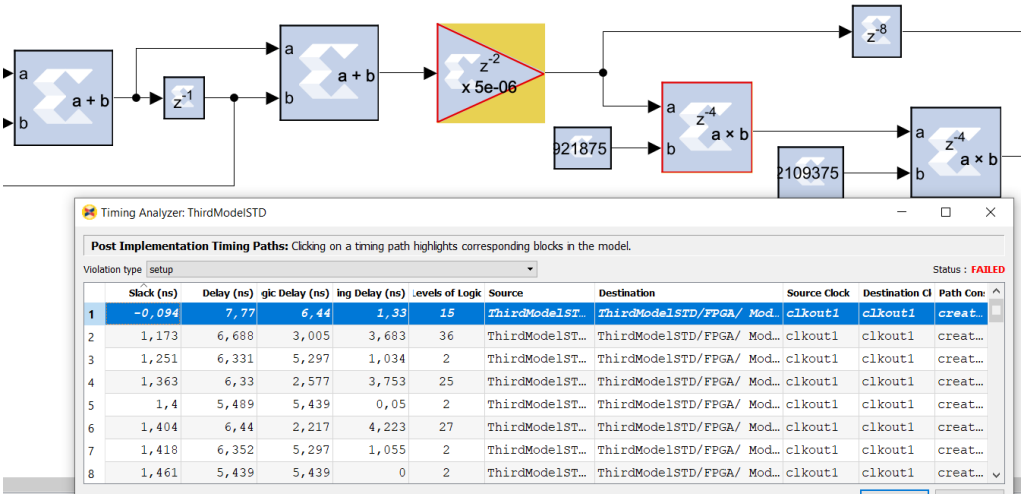


Fig. 81 Timing analysis fail

This is an example where in the multiplication block the time for the operation was greater than 2 ticks, so it was augmented. After all the calibration and fix following the report generated by the failure, another analysis is performed.

```
Updating Model...
Starting System Generator code generation...


Timing analysis finished.
Elapsed time is 00:56:06.
```

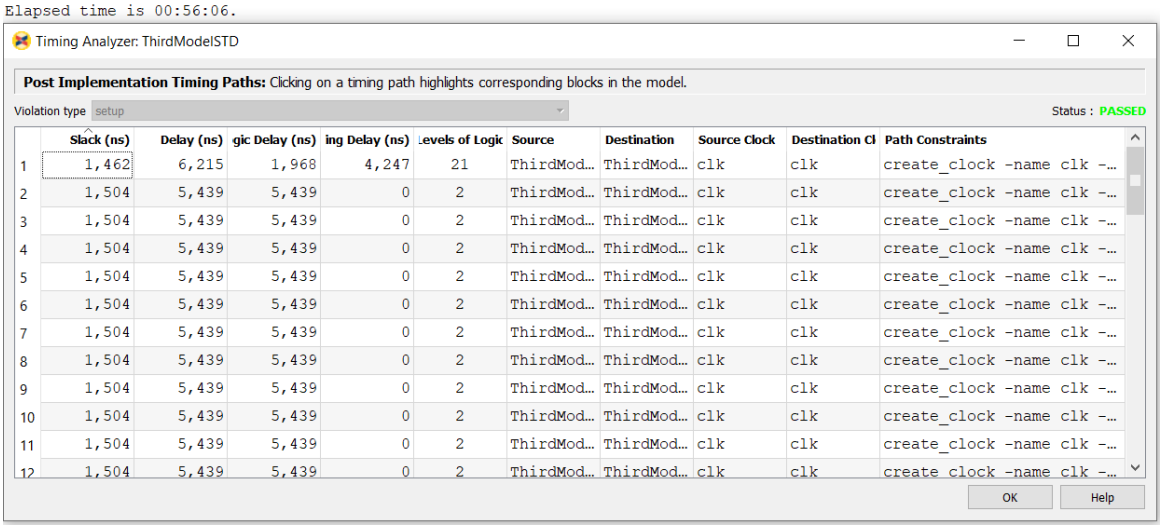


Fig. 82 Timing analysis Passed

Build process

This process is repeated until the report generated is positive, and the model has passed the test. When this is true, we are ready to perform the build that will generate a .ini file needed by configuration desk.

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

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

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

Fig. 83 Build process

The build process is also needed because it's necessary to configure the FPGA blocks for the right board, in our case the DS2655 M1, M1 indicate the first module od the device.

Configuration Desk

The file generated with MATLAB is then exported in Configuration desk, a software from dSPACE for configure the model for the simulator. Other than the FPGA build model it is also needed the processor model and the hardware specification of the HIL simulator.

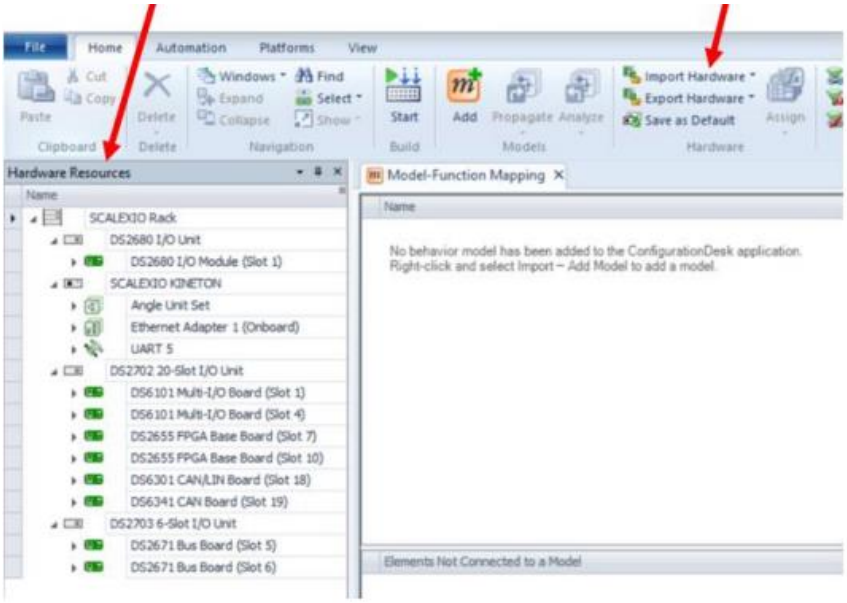


Fig. 84 Configuration desk hardware setting

This tool is also useful for configure the right step time of the processor. When all the check are done and configuration is completed, the las build can be performed to create an .sdf file that will be uploaded directly on the simulator.

Control desk

Finally the file can be uploaded on the HIL simulator through control desk, which is also a graphic interface for the test. So, it is possible to create layers with different tool, graph input variable, displays etc…

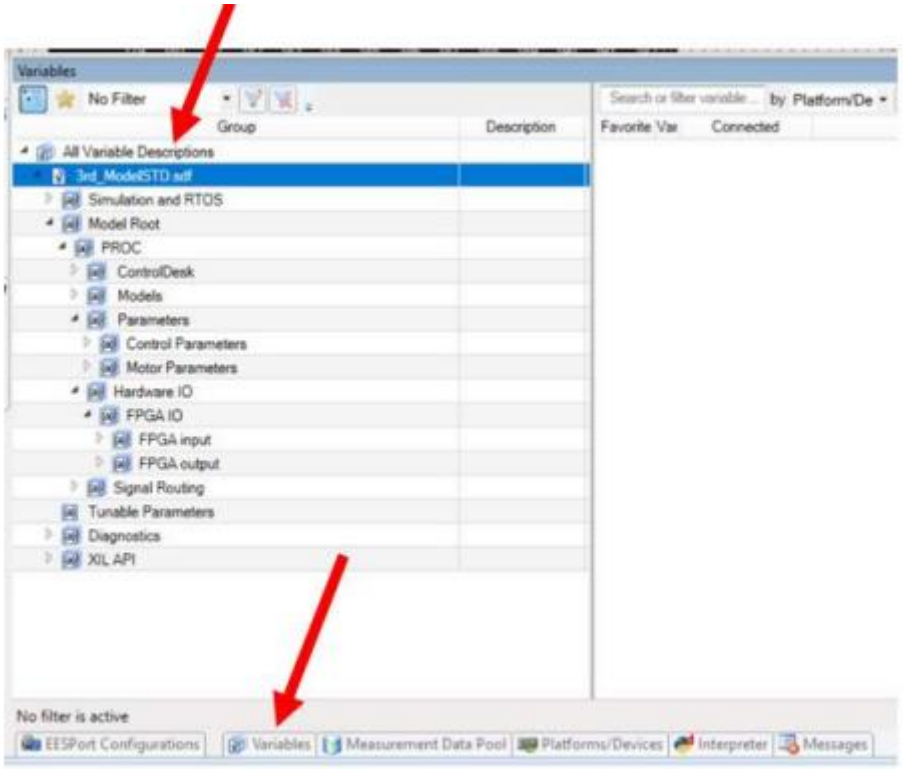


Fig. 85 Control desk Variable setting

At this time the configurator has also to link all the proper variable needed and useful for the test and control to the various element of the layer (graph, knob, etc..).

Result

Now that we have gone through all the workflow of the thesis, it is time to analyse the result obtained with the last testable plant. Before that, in necessary to explain something about the parameters

Parameters

```
%% Phisical Parameters

J  = 0.5;                    % Moment of inertia [kgm^2]
P  = 2;                      % Pairs of Pole
F = 0;                       % Viscour resistance [kgm^2]
Tl = 0.05;                   % Load Torque [Nm]
Pn = 160000;                 % Nominal Po wer [W]
sn = 0.01;                   % Nominal Slip
```

Fig. 86 Physical Parameters

```
%% Electical Parameter

Vdc = 1287;                          % Nominal Voltage
fn = 84;                             % Frequency [Hz]
Rr = 0.103;                          % Rotor resistance [ohm]
Rs = 0.223;                          % Stator resistance [ohm]
Lm = 0.0438;                         % Magnetizing inductance [Hr]
Lls = 0.00158;
Llr = 0.002076;
Lr = Lm+Lls;                         % Rotor inductance [Hr]
Ls = Lm+Lls;                         % Stator inductance [Hr]
sigma = 1-Lm^2/(Ls*Lr);              % Sigma
w = 2*pi*fn*P;                       % Nominal Syncronus Speed [rad/s]
ns = 60*fn/P;                        % Nominal Syncronus Speed [rpm]
Zeq =(Rs+1i*w*Lls)+ Den;              % Impedence
Is= Vdc/Zeq/sqrt(2);                 % Stator Reference Current
Id_ref = real(Is);                   % D-axis Reference Current
Iq_ref = abs(imag(Is));              % Q-axis Reference Current
Iq_max = 300;                        % Current Max
```

Fig. 87 Electrical Parameters

```
%% PI parameters

Kp_speed = 50;
Ki_speed = 500;
Kp_i = 1;
Ki_i = 100;
```

```
%% Clock parameter

T_sample =  1e-3;                   % Processor Sample Time
FPGA_Step = 8e-9;                   % FPGA Sample Time
integral_step = 8e-9;               % Integral Sample Time
T_PWM = 8e-4;                       % Period PWM
Den = 1/(P*Lm^2*Id_ref);            % Attenuation Constant
```

```
%% Inverter

T = 1/(fn);                         % Period of the inverter
RC = 2*T_PWM;                       % Capacitor and lresistor Constant
```

Fig. 88 Controller, inverter, and general Parameters

This were the parameters utilised for the tests, as it can be noticed from the voltage, this is not a motor for automotive, but an industrial one. This choice was made because the lack of complete and stable parameters of other motors, but this again shows the adaptability of the models and how they can change behaviour easily.

Choosing the parameters often gives us troubles, that's because our models are very sensible to the parameters, especially the controller. The difficulty here was to create a robust control that could handle some imperfection from the ideal case, real controllers have lot of control and function to handle every possible case, and this study and design was out of the context for this thesis. This has made the choosing and change oh the parameters a critical issue that could be improved in the future.

Screen

Let’s now see some results obtained from the tests.

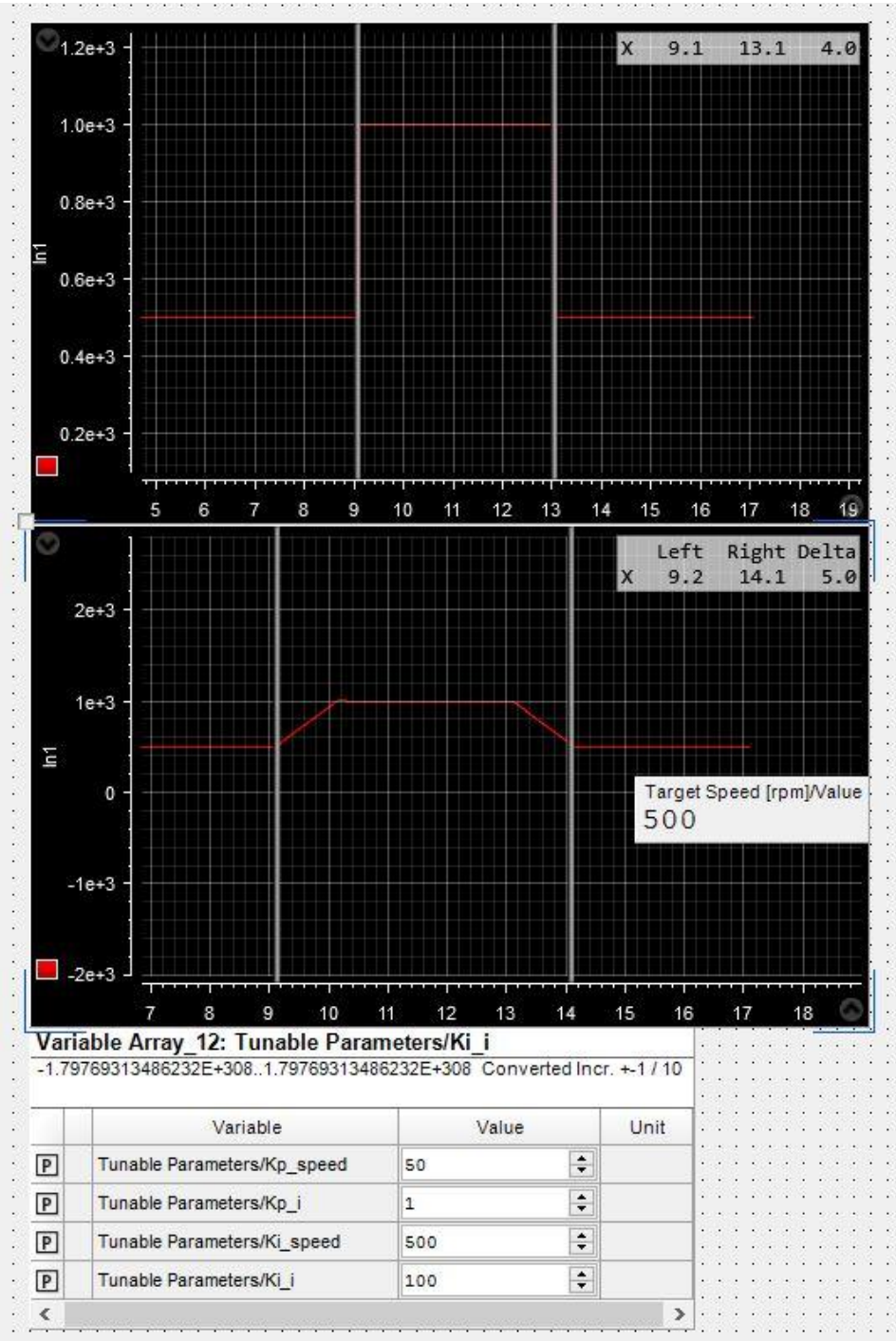


Fig. 89 Rotor speed following target speed

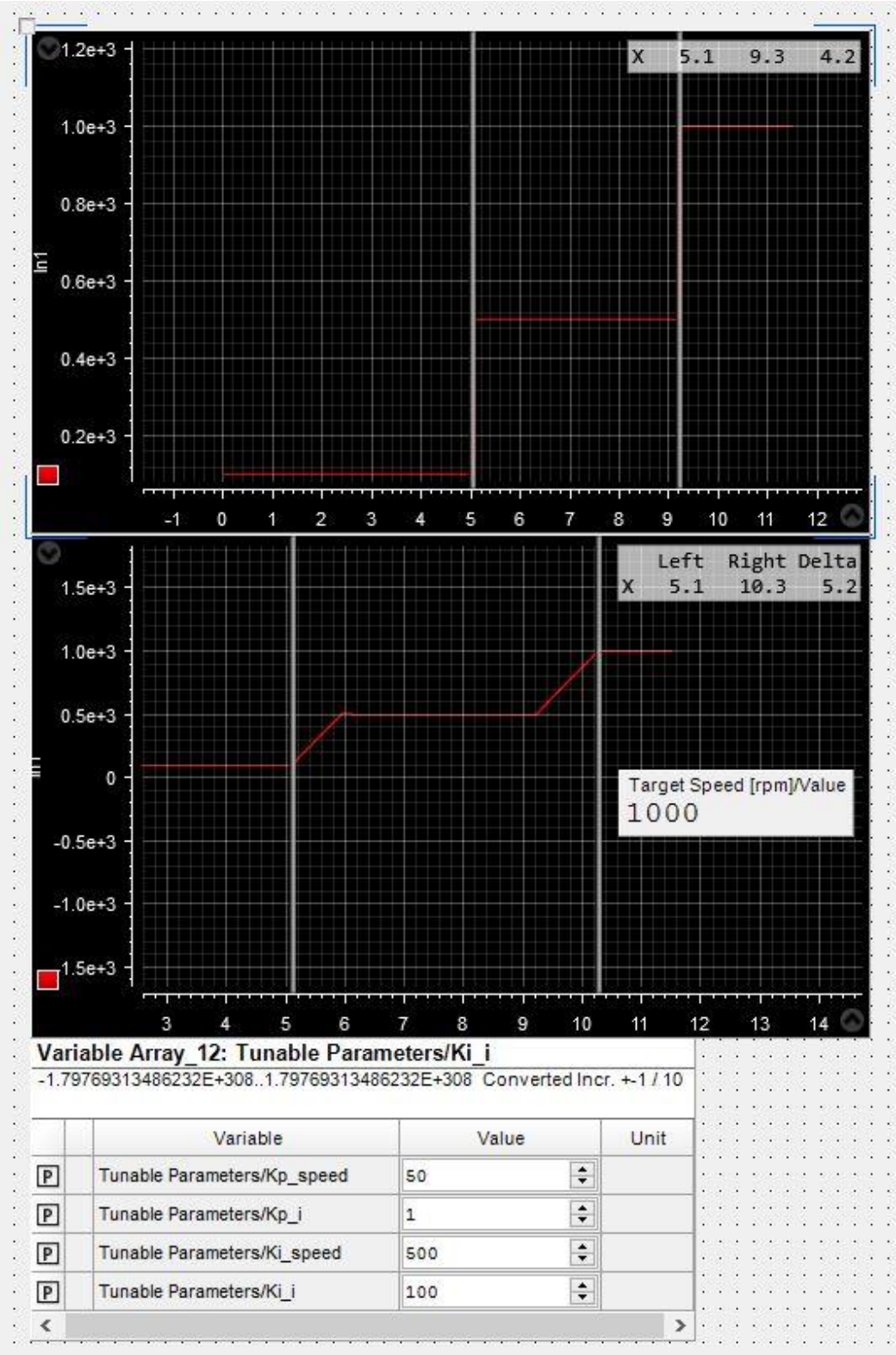


Fig. 90 Rotor speed following target speed

These tow picture above, are result taken directly from control desk, both with the same parameters. The two graphs in the figures represent the input target speed ω_{target} (graph on the top) and the induction motor output speed ω_s (graph on the bottom) and a tab with the control parameters. Obviously the first speed has an ideal instantaneous change, while the real one has some delay to reach the target and some overshoot that will be analysed after. We perform various test to make sure that the response was acceptable, here are reported some acceleration and deceleration.

In the next two picture is shown the overshoot when reaching 1000 RPM. The overshoot is one of the characteristics to be controlled, a very high overshoot is both dangerous and energy consuming. Dangerous because reaching higher speed than requested is for sure a risk and energy wasting because we consume more energy to reach higher velocity and waste it to return at the target. The two graphs represent two different set of parameters that we will call set 1 and set 2, the difference can be seen in the databox on the top right corner, and we have:

- Delta overshoot 1: 7 RPM
- Delta overshoot 2: 2.6 RPM

This means that the set 2 of parameters perform better of the 1, in this specific test, because it is not necessary sure that the set 1 has overhaul better performances.

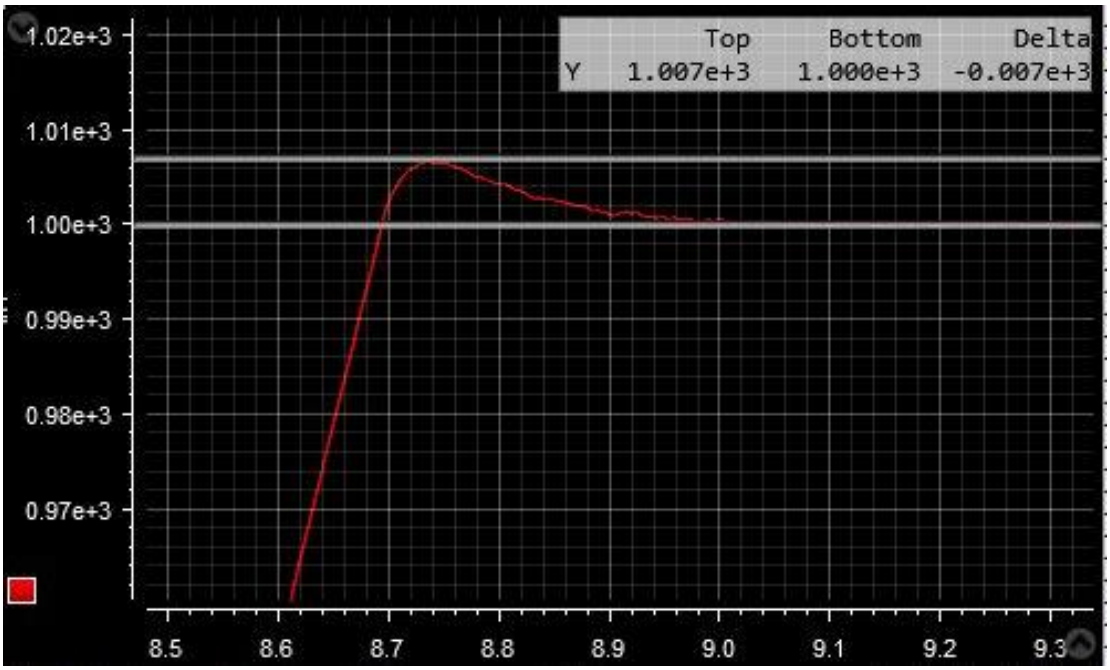


Fig. 91 Overshoot 1

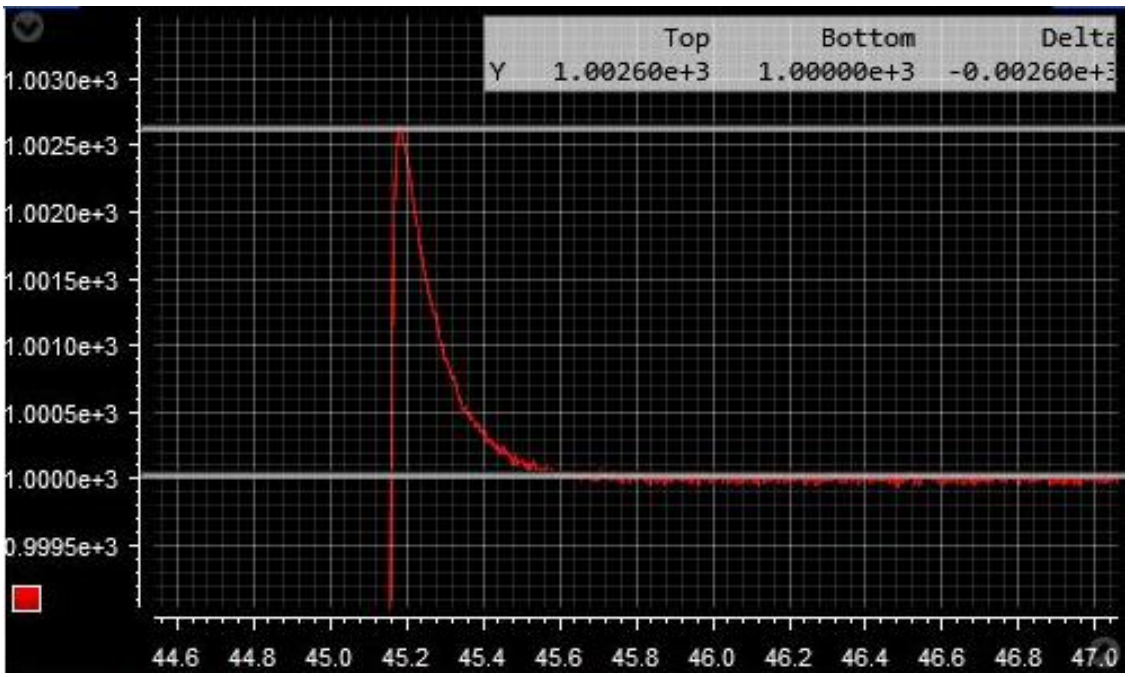


Fig. 92 Overshoot 2

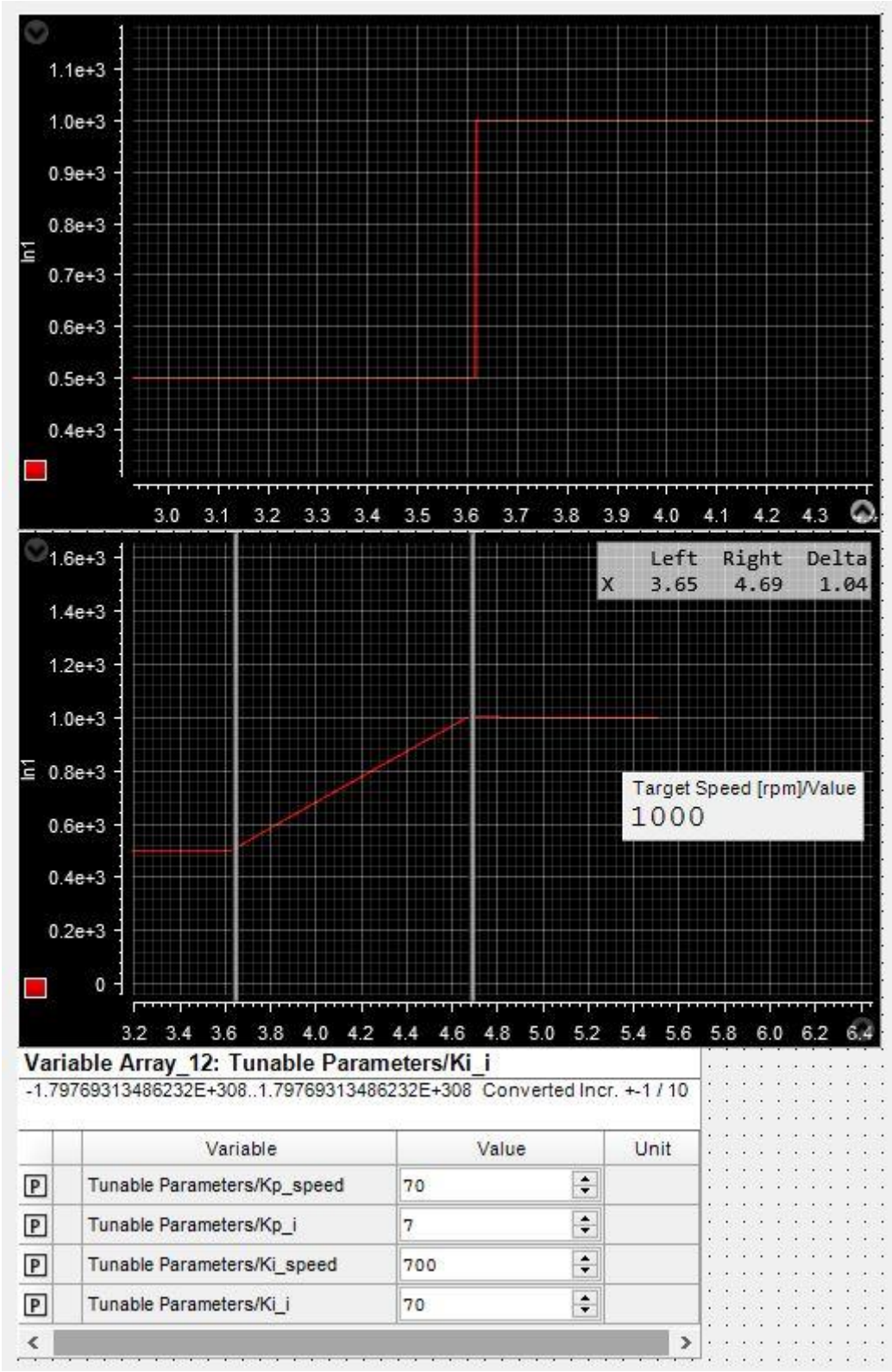
The two sets of parameters are reported in the tabs below.

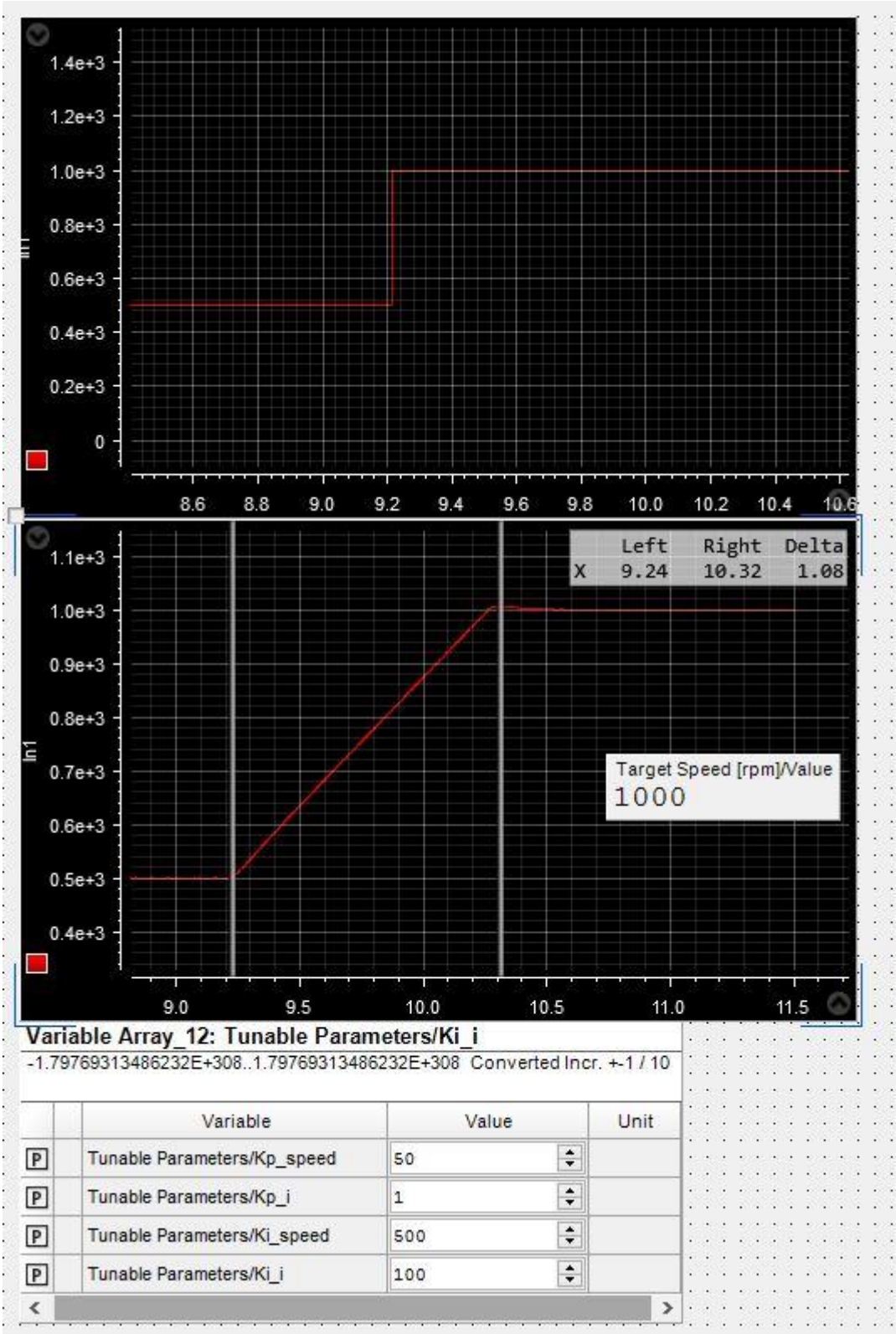
Overshoot 1:

Kp_{speed}	50
Kp_i	1
Ki_{speed}	500
Ki_i	100

Overshoot 2:

Kp_{speed}	70
Kp_i	7
Ki_{speed}	700
Ki_i	70


1.1e+3
1.0e+3
0.9e+3
0.8e+3
0.7e+3
0.6e+3
0.5e+3
0.4e+3
In1
3.0 3.1 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0 4.1 4.2 4.3
Left Right Delta
X 3.65 4.69 1.04
1.6e+3
1.4e+3
1.2e+3
1.0e+3
0.8e+3
0.6e+3
0.4e+3
0.2e+3
0
In1
Target Speed [rpm]/Value
1000
3.2 3.4 3.6 3.8 4.0 4.2 4.4 4.6 4.8 5.0 5.2 5.4 5.6 5.8 6.0 6.2 6.4
Variable Array_12: Tunable Parameters/Ki_i
-1.79769313486232E+308..1.79769313486232E+308 Converted Incr. +-1 / 10
Variable
Value
Unit
Tunable Parameters/Kp_speed
70
Tunable Parameters/Kp_i
7
Tunable Parameters/Ki_speed
700
Tunable Parameters/Ki_i
70


1.4e+3
1.2e+3
1.0e+3
0.8e+3
0.6e+3
0.4e+3
0.2e+3
0
In1
8.6
8.8
9.0
9.2
9.4
9.6
9.8
10.0
10.2
10.4
10.6
Left Right Delta
X 9.24 10.32 1.08
1.1e+3
1.0e+3
0.9e+3
0.8e+3
0.7e+3
0.6e+3
0.5e+3
0.4e+3
In1
Target Speed [rpm]/Value
1000
9.0
9.5
10.0
10.5
11.0
11.5
Variable Array_12: Tunable Parameters/Ki_i
-1.79769313486232E+308..1.79769313486232E+308 Converted Incr. +-1 / 10
Variable
Value
Unit
Tunable Parameters/Kp_speed
50
Tunable Parameters/Kp_i
1
Tunable Parameters/Ki_speed
500
Tunable Parameters/Ki_i
100

All the result shown in the previous section were taken from the internal oscilloscope of Control Desk, which was very help, but has some limitations. The internal oscilloscope could only take measurement from the processor part, so all the simulated components on the FPGA were inaccessible. In order to measure and check the correctness of the behaviour of that components we had to take those signals directly from the pin out of the FPGA on the simulator and analyse them with an external oscilloscope.

Wiring Harness

Thanks to numerous manuals we were able to identify the correct output pins corresponding to the ones assigned from Simulink, the map of the adapter and the corresponding pins is shown below.

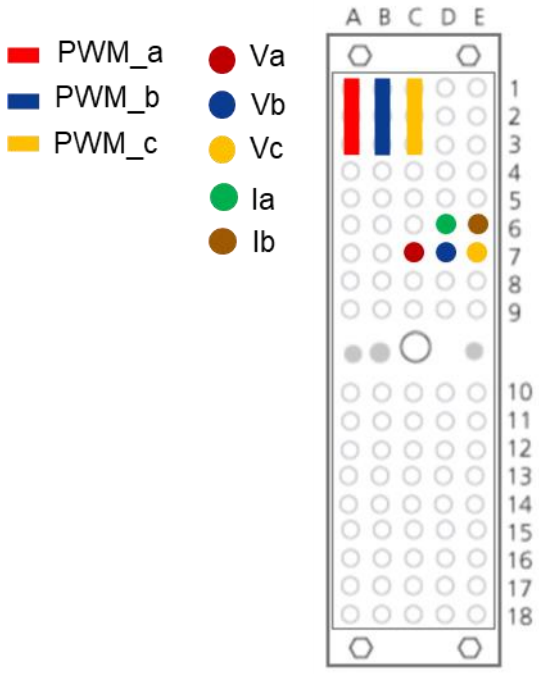


Fig. 93 Hypertac Port Map

Unfortunately the oscilloscope had only two channel, so we weren't able to see all the signal simultaneously

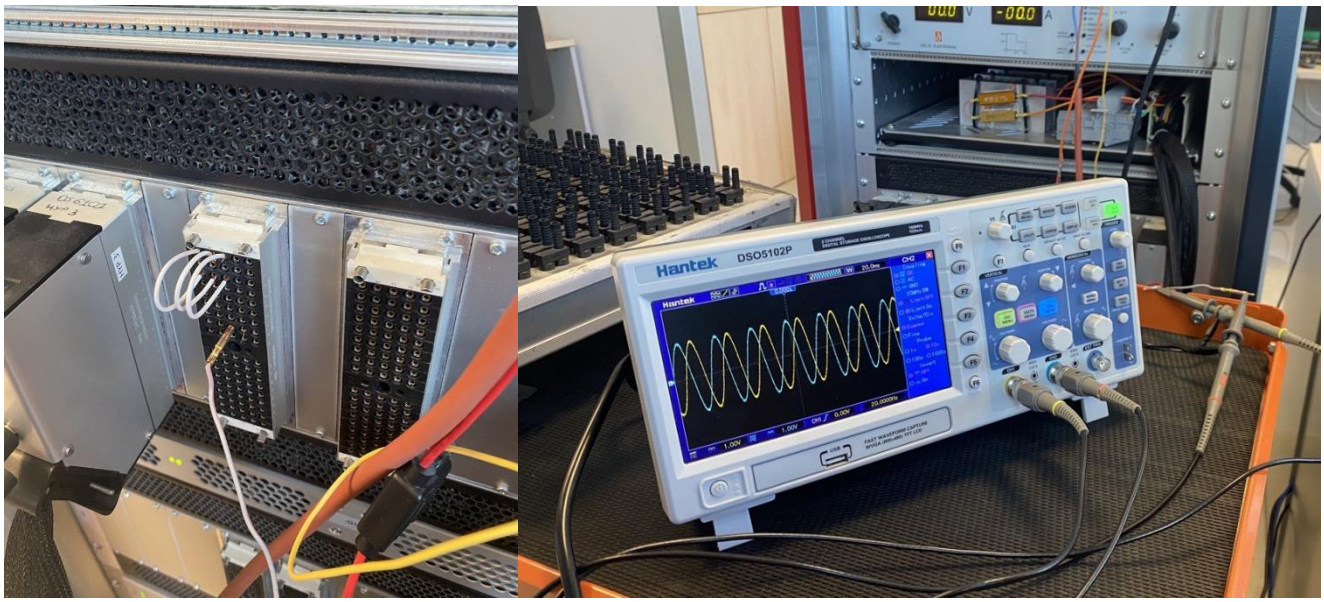


Fig. 94 Wiring Harness

Below are reported also some acquisition from the oscilloscope, showing the PWM of the inverter, and the sinusoid of the voltage V_{ab}

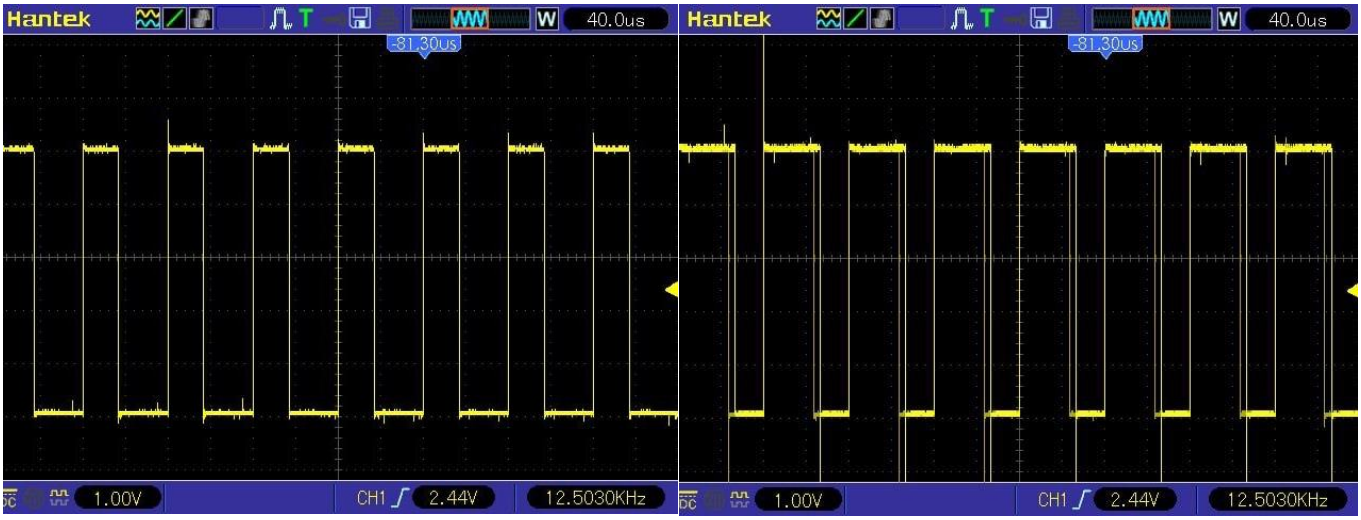


Fig. 95 PWM

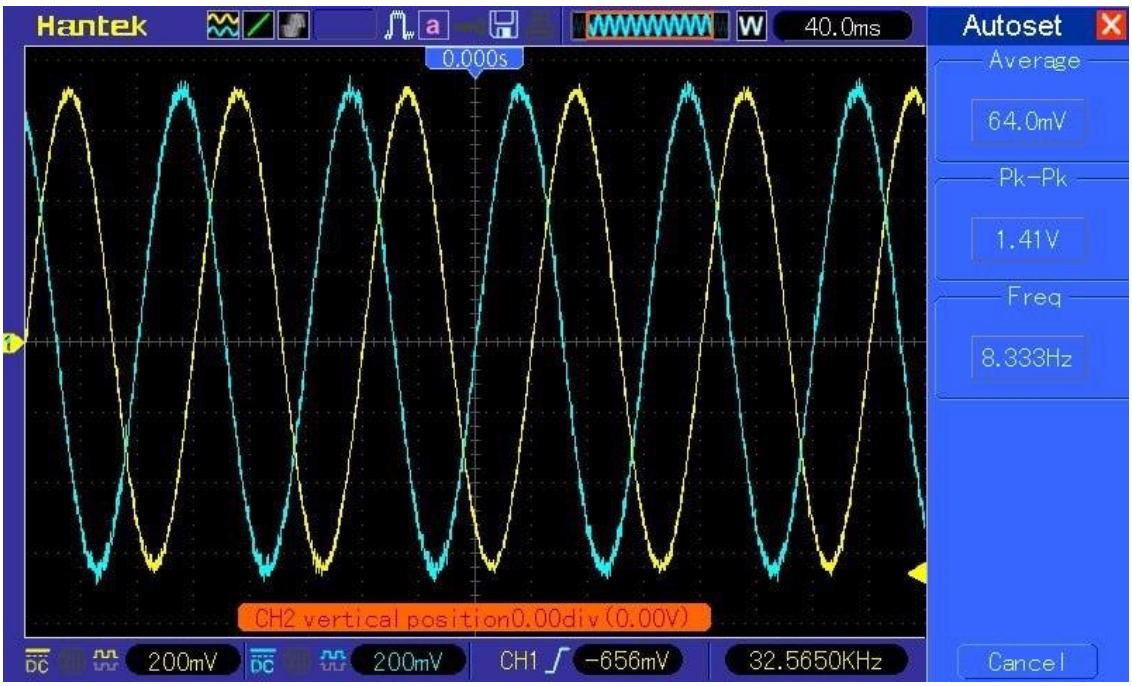

Hantek
40.0ms
Autoset
Average
64.0mV
Pk-Pk
1.41V
Freq
8.333Hz
0.000s
CH2 vertical position0.00div(0.00V)
200mV
200mV
CH1
-656mV
32.5650KHz
Cancel

Fig. 96 Vab

Future Development

In this final chapter oh the thesis we will focus on the possible change and improvement of the model, to enhance its reliability, robustness and to make the tests more comparable with the real ones.

There could be add lot of component to improve the complexity and make the system more similar to the real ones, such as:

- **Battery**, the model of a high voltage battery to power the inverter and the motor, also with its related problem like overheating.
- **Gearbox**, to make the motor work always at the optimal speed to ensure lower battery consumption and greater efficiency.
- **Frictions,** for this model the friction and mechanical loss weren't taken in account, but with an addition of a battery, it will become very important.

Other change could be the improvement of the already existing components like the inverter, which is an RC inverter, not used for automotive application and implement a model of the switch of a real inverter also adding all its feature and non-idealities like electrical losses.

Also the motor could be improved by adding a feature to make it work as a generator. In fact during the deceleration the motor could be used as a brake wit the so called regenerative braking that transform the inertia of the vehicle in mechanical movement an then in electrical power. This would be more important with the addition of the battery, because the electrical power generated could be used to recharge the battery.

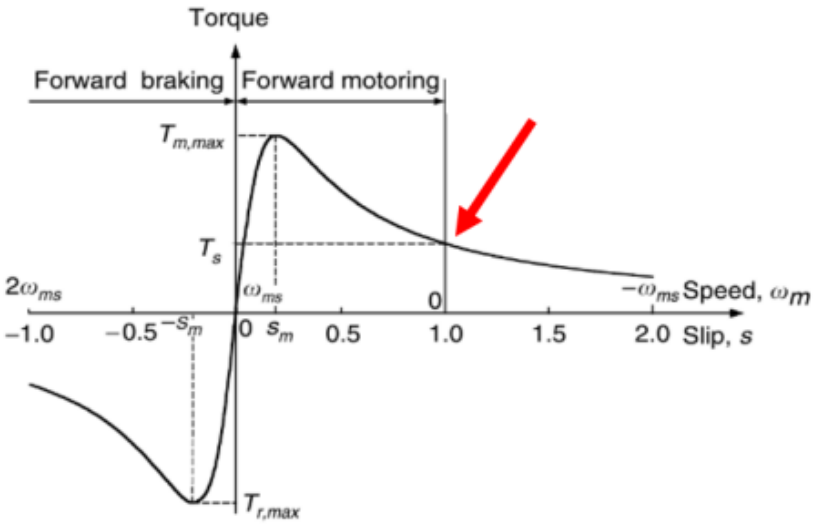


Fig. 97 Regenerative Braking

Another next step could be also to study the impact of different hardware architectures on Hardware in the Loop systems, or investigate the impact of different algorithms on the accuracy and efficiency of the simulations and tests.

Thanks

I would like to take this opportunity to express my gratitude to all those who have contributed to the completion of this thesis. First and foremost, I am deeply indebted to my thesis advisor prof. Alessandro Rizzo, and the company advisor eng. Antonio Vitale whose guidance and support throughout the entire process has been vital for this work. The experience, feedback, and encouragement have been critical to its success.

I would also like to thank my colleague Fabio who shared with me the thesis process, supporting me, and giving precious advice.

A heartfelt thanks goes also to my love Alexandra, for the support and encouragement throughout the duration of this thesis. Her patience and kindness help me a lot during the difficult times.

Furthermore, I would like to acknowledge the support of my family and friends, who have provided unwavering encouragement and understanding throughout my academic journey. Their love and support have been a constant source of inspiration and motivation.

Finally, I extend my heartfelt thanks to all the participants who gave their time and shared their insights for this study. Without their contributions, this thesis would not have been possible.

Thank you all for your contributions and support. I am honoured to have had the opportunity to undertake this project.