Università degli Studi di Padova

Facoltà di Ingegneria

Corso di Laurea Magistrale in Ingeneria dell'Automazione

Master's thesis

Validation Platform for Grid Forming Control Strategies of Power Electronic Inverters

From Component to System Level Validation

Supervisor: Luca Schenato
Co-supervisor: Cătălin Gavriluță, Adolfo Anta, Ali Tayyebi-Khameneh

M.Sc.: Denis Vettoretti

2 December 2019

Abstract

In the last years green energies have become more and more relevant in the scenario of energy production, also thanks to European politics. The transition towards a 100% renewable system have introduced new challenges correlated to the grid stability and reliability. In order to address all these issues, novel grid forming techniques has been proposed. Several studies present promising results, mainly based on offline simulations. However, despite the active research effort in the field, at the moment, no real-time simulations, as well as, laboratory prototypes which implements grid forming control strategies have been performed or tested. Due to the lack of working prototypes and practical applications, the use of these novel strategies are still uncertain. More than analysing the behaviour of the grid forming techniques, this thesis attempts to give a contribution by providing a methodology for performing controller-hardware-in-the-loop simulations. In particular, two different platforms are configured in order to perform component and system level validations. Several technical solutions are presented to guarantee electrical compatibility between different parts of the setups. Furthermore, apposite libraries and automatic routines are developed to facilitate the user during the simulation process. In order to expand the flexibility of the platform used to perform system level validation, a fully configurable interface board is developed and tested. The proposed platforms are fully customizable to allow investigations of other test cases different from the examples proposed in this work.

AIT Austrian Institute of Technology is Austria's largest Research and Technology Organisation (RTO) and an international key player in many of the research areas it covers. This makes AIT a leading development partner for the industry and a top employer within the international scientific community. [1]

Acknowledgment

I would like to thank my supervisor Luca Schenato from the University of Padua to have offered me the great possibility to write this thesis. I would also like to thank the Austrian Institute of Technologies (AIT) that welcomed me and gave the possibility to use its resources without limitations. I found very good colleagues, always ready to help me. In particular a huge thank goes to my supervisors Cătălin Gavriluță, Adolfo Anta and Ali Tayyebi-Khameneh, that help me every single day during this path. Cătălin was not only a supervisor to me, but also a friend and a mentor that pushed me forward day by day. Thank to him I learnt that there are no problems, only challenges. Cătălin let me explore all the fields of knowledge I'm interested in, thus giving me the chance to improve my abilities. With regard to this aspect I would also like to thank Biswas Sumanta (Anto for friends) who shared his knowledge about PCB prototyping with me. And now let me switch to Italian to thank my family. Vorrei ringraziare la mia famiglia che mi ha supportato in questo percorso di studi offrendomi ogni tipo di sostegno possibile. Mia madre Barbara per aver sopportato tutte le volte che le ho messo sottosopra casa per i miei progetti, per avermi lasciato introdurre ogni tipo di tecnologia e avermi sostenuto durante i periodi difficili. Grazie mamma! Vorrei poi ringraziare mio padre Daniele che mi ha sempre spinto fin da piccolo a migliorarmi e mi ha proposto nuove sfide. Grazie a lui ho potuto fin da giovane utilizzare attrezzature e sviluppare progetti introducendo nuove tecnologie e macchinare nell'azienda di famiglia. Grazie papà! Vorrei poi ringraziare mio fratello Michael per avermi sopportato durante tutte le sessioni di esame lasciandomi i miei spazi e rinunciando lui alcune volte alle sue necessità per dare spazio alle mie. Grazie Michael! Vorrei poi ringraziare i miei nonni Giovanni ed Eulalia per tutto il supporto che mi hanno dato. Sono loro che mi hanno cresciuto e sempre sostenuto quando i miei genitori erano a lavoro. Grazie nonni! E infine vorrei ringraziare Francesca, la mia ragazza, che mi ha sempre spinto ad affrontare nuove sfide e ad intraprendere nuovi viaggi. Lei sa quanto sia stato difficile questo percorso per me e mi è stata d'esempio perché lei è il mio esempio. Lei mi ha insegnato a reagire sempre, a non mollare mai e a darmi da fare come si deve. Mi ha poi aperto nuovi orizzonti, ha creduto in me, come tutta la mia famiglia, quindi sebbene a parole non sia bravo a esprimermi spero di poterglielo dimostrare con i fatti. Grazie Francesca!

Contents

Chapter 1

Introduction

This chapter provides an introduction to the problems and challenges of a low-inertia grid caused by a transition toward a 100% renewable energy based system. Since the European Union politics is encouraging the development and use of renewable energy sources, the electrical network is undergrounding a drastic change. To support this transition, novel control strategies are under development and test. The objective and the contributions of this thesis are presented at the end of the chapter.

1.1 Background

One of the most important challenge of the last years is to limit the global warming. This phenomenon is largely caused by the transport industry and the use of fossil fuels for the production of electric energy. In fact, the electrical energy demand is mainly covered by fossil fuels or gas power plants. The usage of these non-renewable sources inevitably leads to the production of CO_2 and other greenhouse gases which contribute to global warming. The impacts and the risk of climate change are well known and for that reason Europe is working on several fronts to address the problem. In particular, with the Paris Agreement (UNFCCC, 2015) [2] all European countries undertake to respect the agreements adopted in order to reduce the consumption of fossil fuels and to increase the deployment of renewable energies. The European Union set a mandatory 20% share of renewable energy for the 2020. Figure 1.1 shows the progress reached by each European country in a period from 2005 to 2016.

Thanks to the introduction of renewable energies Europe is reducing drastically the consumption of fossil fuels, limiting in part the global warming and the planet air pollution. The diagram in Figure 1.2 reports the absolute fossil fuel usage reduction expressed in million tonnes of oil equivalent (Mtoe) and the relative reduction of fossil fuel usage. The relative reduction of fossil fuel usage is expressed as the absolute reduction over the total consumption of fossil fuel.

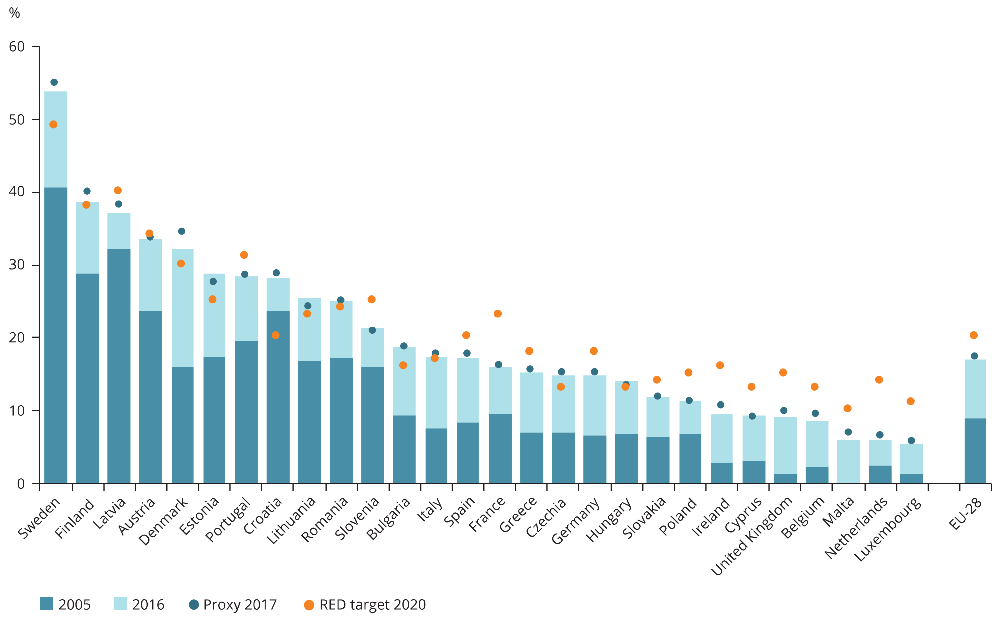


Figure 1.1: The renewable energy shares until 2005 are marked in dark blue, the progress obtain the following years until 2016 are marked with light blue bars. The orange dots define the target for each country for 2020. [2]

An important aspect to consider is the contribution given by the different renewable sources. Hydroelectric, solar, wind, bio-gases, and geothermal power plants present different growing curves, as shown in Figure 1.3. The hydro-power contribution is pretty constant and this is due to the fact that the majority of the sites suitable to install hydro electric plants are already used. Solid biomass and bio gases present a low increment during the years, while solar and wind sources presents a fast growth. Wind power became the most important renewable energy source in the period 2007-2016 with 37.2% of the total renewable energy production. The solar power growth exceeded the geothermal energy growth in 2008, reaching 119.5 TWh in 2017. In this decade the contribution of the solar power represented the 12.3% of the total renewable energy generation.

It is expected that in the coming years wind and solar energy will become the predominant renewable energy sources, introducing new challenges in the energy transmission and distribution system. Nowadays, the European network is mostly based on centralized generators, i.e. power plants situated far from the end-users and connected to the high-voltage transmission lines. The electrical power grid distributes the electricity to the end-users [3]. The introduction of renewable energy is changing radically the structure of the electric grid leading to a network composed of distributed generators. The high power centralized

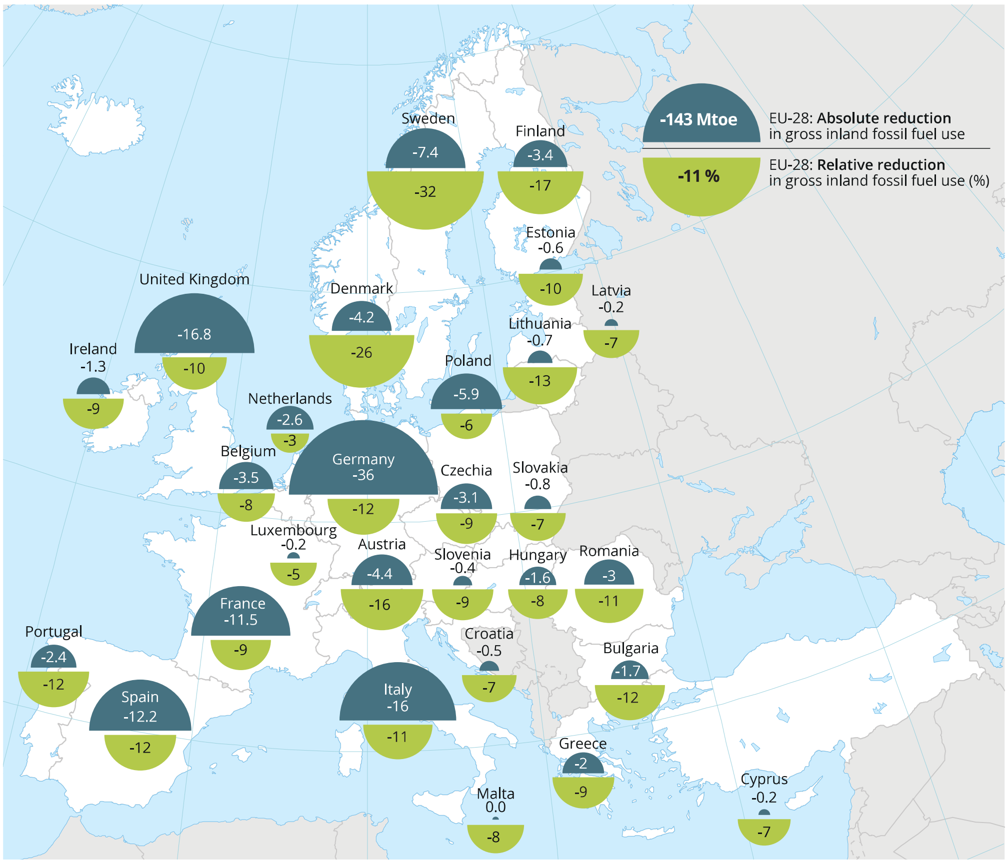


Figure 1.2: Total and relative reduction fossil fuel usage [2].

generators are substituted with small-scale power plants situated near the end-users. The loss of the centralized power plants typically based on synchronous generators introduce new issues that compromise the grid stability.

1.2 Impact of wind and photovoltaic penetration in the electrical transmission network

The centralized power plants based on the synchronous generators present a high rotational inertia that ensures the grid stability. On the contrary, the inertia of the distributed generators based on PV panels and wind turbines is very small or completely absent. Hence, an increase of the photovoltaic and wind penetration introduces a series of issues related to the grid stability. The intermittent nature of these sources can cause oscillations in voltage and frequency of the power systems. In particular, a high level of photvoltaic penetration leads to issues related to power quality, power unbalance between generation and demand, and

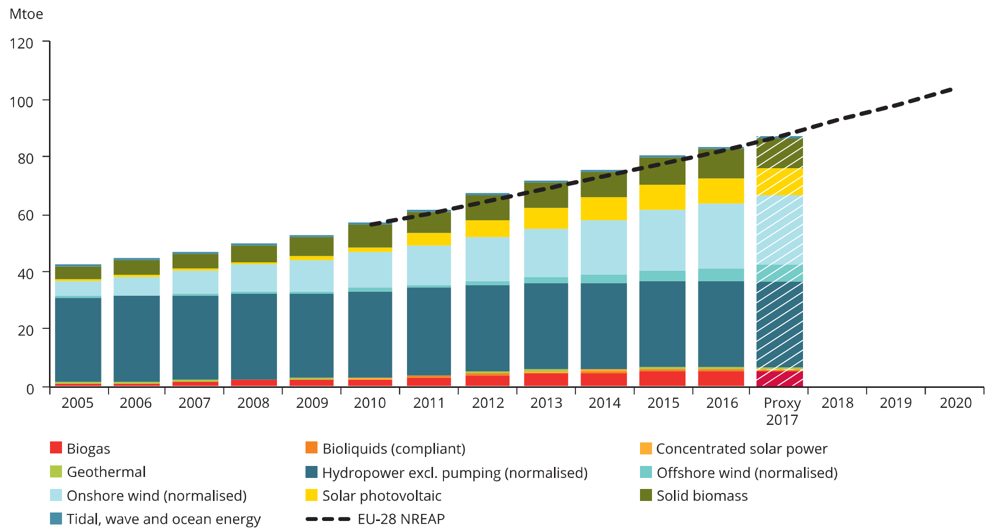


Figure 1.3: Contribution given by the renewable energies expressed in Mtoe in the last ten years [2].

voltage and frequency variations, as reported in [4]. The grid can present voltage fluctuations due to the alternations of clouds and sunshine that change the power production of the PV panels. Furthermore, if the penetration level is high, the resulting power not only compensates the load power, but also causes reverse power flow into the network, which can lead to voltage rise. There are many possible solutions to limit the phenomena, such as the use of capacitors, battery storage systems, etc. A high photovoltaic penetration level in the low level distribution network also causes voltage unbalance. In particular, in a three phase system, voltage unbalance, i.e. difference in the voltage level of the three phases or a difference in the phase angle between two phase voltages, can cause large problems to the electronic devices.

Another important consequence of a high photovoltaic penetration is the impact that it has on the frequency response of the transmission system. When the sun irradiance decreases, the power capability of the PV farms drops. With a network composed mostly of synchronous generators, it is assumed that there is always enough fuel to feed the turbines. In a grid where a consistent percentage of the energy produced comes from PV panels, if some areas have a low irradiance, the power capability of the renewable power plants decreases and may not satisfy the energy demand. In this scenario an imminent consequence is the decrease of the grid frequency. A deviation from the reference grid frequency (in Europe 50Hz) leads to efficiency problems and may cause the damage of the synchronous generators. The frequency drop occurs because the renewable power plants use a grid following approach. The synchronous generators are responsible for forming the grid. In fact, the synchronous generators form the grid frequency, control the voltage, compensate power imbalance, etc. Meanwhile, the power plants based on renewable energies, which are controlled by a grid following strategy just inject power into the network. Therefore, the grid stability is ensured by the synchronous generators.

A comprehensive study on the effect of the PV penetration on frequency response is presented in [4]. In the study the focus lies on the frequency response due to variation of PV penetration when the loss of a synchronous converter occurs, as shown in Figure 1.4. With a low penetration the frequency droop is small. On the contrary, as soon as the penetration level increases, it leads to a large frequency droop that can cause grid instability or the damage of the synchronous generators.

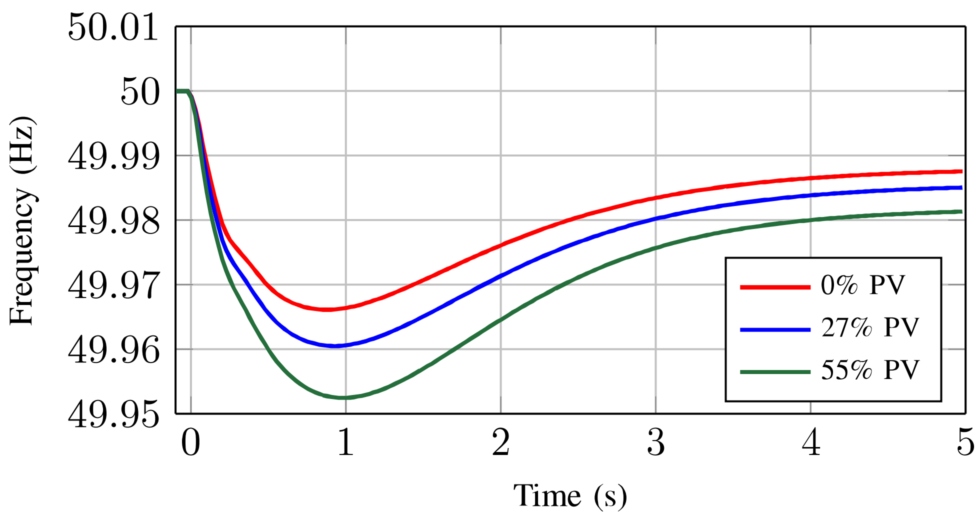


Figure 1.4: Frequency response at different level of PV penetration using grid following techniques [4].

In the same way, as the PV panels, the wind farms present the same intermittent operation mode due to the wind conditions and are also controlled using a grid following strategy. Therefore, power imbalance and voltage fluctuations must be taken into account.

Overall, the increase of renewable energies, in particular based on solar and wind sources, are drastically modifying the grid structure and dynamic. A low inertia grid based on distributed generators is catching on and for this reason it is necessary to address all the problems that such a grid involves. To allow the transition toward a 100% renewable based grid, new strategies must be adopted in order to substitute the conventional synchronous generators. With the increase of renewable energy penetration it is no longer possible to delegate the grid stability, the frequency response, the power imbalance compensation, etc., to the synchronous generators, but it is necessary to implement new strategies, the so called "grid forming techniques", to control the electrical grid of the feature.

1.3 State of the art

In the last years, novel grid forming techniques, such as droop control, virtual synchronous machine, matching control, and dispatchable virtual oscillator [5] [6] [7] [8] have been developed to address the transition toward a 100% renewable grid. These new strategies try to mimic some aspects of the synchronous machines and they have the purpose to substitute the synchronous generators by ensuring the grid stability, even if the network presents a low inertia. In the literature it is possible to find several studies about the grid forming techniques where the advantages and disadvantages of each of them are taken into account. In general, to ensure the stability of a low inertia network, the grid forming strategies present a faster primary frequency response with respect to the synchronous generators when a load disturbance occurs. Furthermore, it was observed that all of the control strategies are able to synchronize to the grid and to form the grid in case of the loss of synchronous generators. It has been demonstrated that the matching control is asymptotically globally stable while the other techniques have been validated through offline simulations. Different scenarios have been considered to perform the simulations, e.g., multiple grid forming converters connected through the network or in island operation. Moreover, a performance comparison that highlights the interaction between a synchronous machine and the grid forming converters has been proposed in [8]. The response of the novel strategies to large changes in load and loss of synchronous generators has been analyzed and represents the state of the art [8]. Currently at *AIT Austrian Institute Of Technology* we are studying the grid stability at different penetration levels in order to evaluate the behaviour to the novel grid forming techniques and highlight their strengths and weaknesses. Figure 1.5 presents an equivalent scenario of Figure 1.4 but in this case they adopted grid forming control strategies instead of a grid following strategy. The frequency droop decreases with the increase of the renewable energy penetration highlighting the benefits introduced by the novel techniques.

1.4 Objective and contributions of this work

The literature about grid forming techniques is plentiful, many studies about the stability of the grid and the interaction between the grid forming based converters and the synchronous generators have been made. To validate the novel strategies, the scientific community mostly appeals to offline simulations. However, real-time simulations of the grid forming strategies lack, as well as

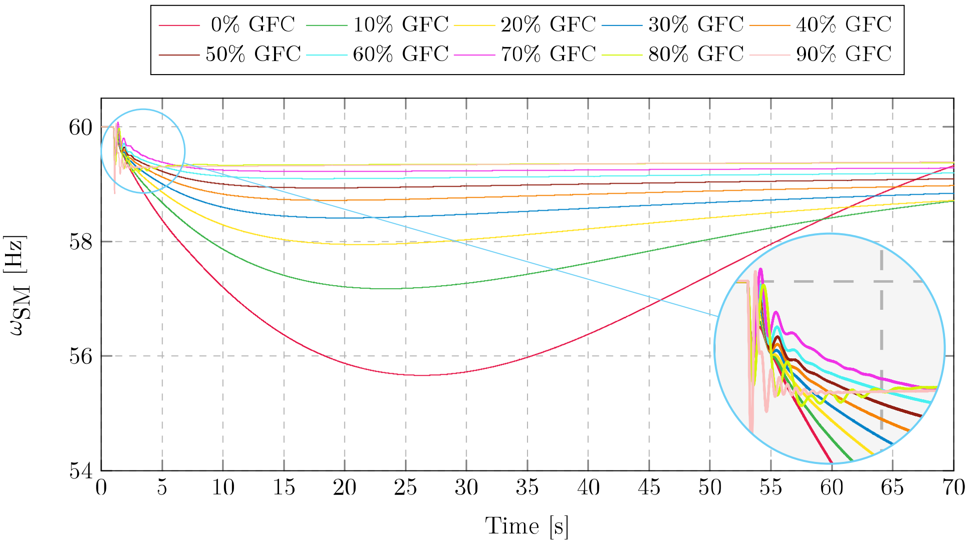


Figure 1.5: Frequency response at different level of PV penetration using grid forming techniques [9].

their implementation in real hardware. This kind of simulations and tests are very important to increase the validation accuracy, to establish if the control strategies are robust enough to guarantee the grid stability, and to address all the problems introduced by the electrical hardware limitations.

The purpose of this thesis is to build and configure two setups able to perform component and system level validation of the grid forming techniques. We will introduce and describe the hardware used in the two different setups, explaining how to configure all the required parameters and then we will provide a possible methodology to follow in order to perform controller hardware in the loop simulations. First of all, we will focus on component level validation. We will implement four grid forming techniques, described in Chapter 2, in a control card connected to a real-time simulator that emulates a simple network. In this way, we will test the control card and the behaviour of the grid forming based controller implemented in it. At the end of Chapter 3 the user will have the knowledge needed to configure the setup and to perform real-time simulations.

Secondly, we will focus on system level validation, and we will present a new and more high performance setup. With it, it is possible to simulate the network of some regions or countries in real-time in order to study the behaviour of the grid at different penetration levels and the interaction of multiple grid forming converters with the conventional synchronous generators.

Overall, this work aims at promoting the development and the implementation of the novel grid forming techniques in real hardware. We will try to provide a "tool" aimed at pushing the research a step forward and at making the novel strategies a bit closer to the commercial world.

Chapter 2

Modern Control Techniques for Power Electronics

In this chapter we present four novel grid forming techniques: Droop Controller, Virtual Synchronous Machine, Matching Control, Dispatchable Virtual Oscillator. We start the introduction with the primary frequency control of synchronous generators, explaining its salient features. Subsequently, we compare each grid forming techniques with the primary control model of the synchronous generator, highlighting similarities, advantages, and, disadvantages. [1].

2.1 Introduction

The electrical power system is undergoing an important transition from centralized generation to distributed generation due to the increase of renewable energies, mainly based on wind farms and PV parks [5]. In the early stages, the aim of wind and solar power generators was to maximize the power injected into the grid. Usually, the control techniques used for this purpose are based on a grid following approach. A converter follows a grid following strategy if its controller is designed for a stiff grid and it injects power to the grid at the ac grid frequency usually measured through a phase-loked loop (PLL) [6]. In this scenario the grid stability and the power balance is controlled by the conventional generators that compensate the power fluctuations introduced by the renewable power generators. This approach can be used as long as the percentage of renewable energies with respect to the total installed generators is low. However, in the last years the contribution of the renewable energies is rapidly increasing. Moving towards

[1]**NOTE:** Since this project aims to test and validate the grid forming techniques presented in *Interactions of Grid-Forming Power Converters and Synchronous Machines - A Comparative Study (Authors: Ali Tayyebi, Dominic Groß, Adolfo Anta, Friederich Kupzog and Florian Dörfler)* [10] wrote by researchers working at the Austrian Institute of Technology, I strictly limit the discussion to the models presented in this paper.

a nearly 100% renewable system leads to major consequences, especially due to the loss of the synchronous generators [8]. Therefore, the major challenge is to find a way to control the converters in a distributed power grid [5], addressing the key problem of low-inertia network stability under the loss of the rotational inertia. In comparison to the conventional centralized power plants based on synchronous machines, the new distributed generators have a low or absent rotational inertia. Rotating synchronous generators thanks to their stored kinetic energy add rotational inertia to the grid, which is a crucial property to guarantee damped frequency dynamics and stability. Novel control techniques have been developed to address the challenges introduced by the distributed generators; the so-called grid forming strategies. They are based on the idea of emulatin part of the functionality of the synchronous machines, in order to gradually enable the transition towards a 100% renewable-based generation system. In this chapter we analyze four main grid forming strategies: Droop Control, Virtual Synchronous Machine, Matching Control and Dispatchable Virtual Oscillator Control.

2.2 The basic idea behind the synchronous generator

Synchronous generators are usually controlled by an automatic control system composed of the primary and the secondary frequency control. In certain occasions the system disposes also a tertiary control that usually is activated manually. The tertiary control is used for economic off-line optimizations and does not play a fundamental role in the stability of the grid.
The main task of the primary control is to keep the frequency of the synchronous generators at the reference set point and to quickly respond to deviation from it. The primary control is usually implemented locally at the power plant in order to regulate the valves, gates, or the servos of the power generators. Since the primary control is based on a proportional control law, the secondary control acts at a later time to correct the remaining frequency errors, but also to balance the power flow, in order to correct active power imbalances. A basic block diagram of the primary and secondary control is shown in Figure 2.1. While in Figure 2.2, the temporal structure of the control system is shown. The two main reasons to keep the frequency stable are: a too low frequency can lead to the generation of vibrations in the turbine of the generator provoking its damage [11]. Many of the devices connected to the network work optimally at the nominal frequency. Therefore, the delivered electrical energy quality is lower if the frequency is not at its nominal value. The European grid has an automatic protection system that continuously monitors the grid frequency. If the network goes under a certain frequency the protection system will be activated in order to protect the system.

From the discussion above it emerges why the primary frequency control is so important for the stability and security of the grid. Let's now introduce the

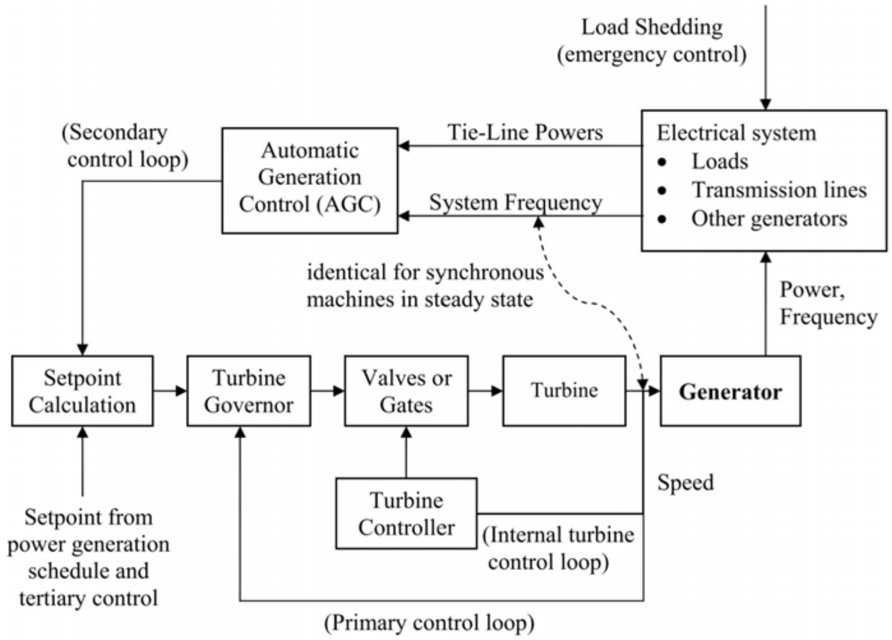


Figure 2.1: Primary and secondary control of a power plant based on synchronous generators [11]

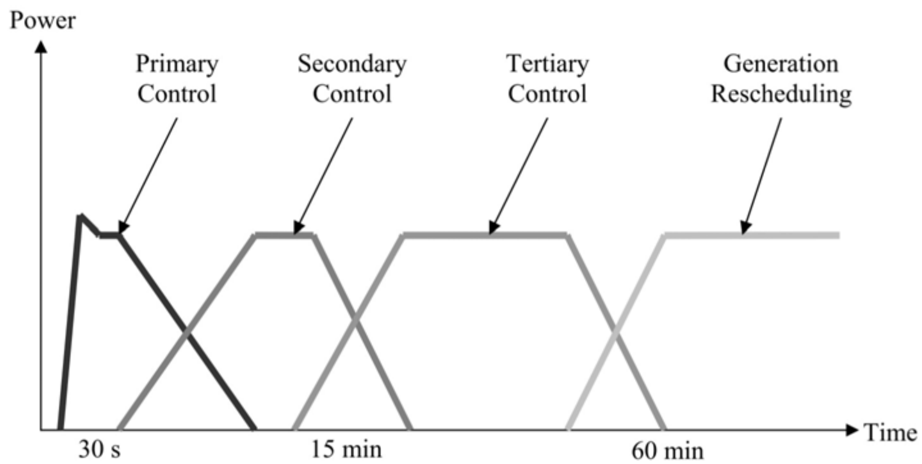


Figure 2.2: Activation time of the different control layers after a disturbance [11]

equations that describes the primary frequency control of a turbine. The control law is a proportional feedback control. An affine relation between the measured frequency and the power plant is established from the controller and it is shown in (2.1).

$$(f_0 - f)d_\omega = \Delta p \tag{2.1}$$

Here, d_ω is the droop gain and Δp is the difference between the power set-point p^* and the governor power output p. The turbine dynamics instead are described in (2.2).

$$\tau_g \dot{p}_\tau = p - p_\tau \tag{2.2}$$

Here, τ_g is the turbine time constant and p_τ the turbine output power [11].

A block diagram representation of the primary control is shown in Figure 2.3. The reference frequency is compared to the turbine frequency and the error feeds the proportional controller, which is characterized by the droop gain d_ω. The output of the proportional controller represents the power amount needed to compensate the frequency error. This term is added to the power set-point and given as input to the internal turbine controller. The turbine dynamics is a first order system represent by (2.2).

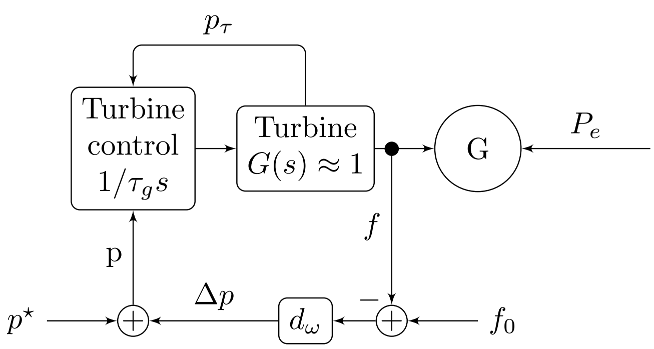


Figure 2.3: Block diagram of the primary control and the turbine dynamics

Based on the primary frequency controller, the novel grid forming strategies try to emulate the behaviour of the synchronous generator, introducing new advantages and disadvantages in the process. In the next sections, all four grid forming strategies will be analyzed, describing the similarities with the primary frequency control of the synchronous generators.

2.3 Droop Control Technique

One of the most popular primary control employed in the grid forming converters is the droop control. It emulates the speed droop property of the synchronous generators. The droop strategy controls the output frequency ($f = 2\pi\omega$) of the voltage source inverter according to the active power of the generator, as shown in (2.6).

$$\dot{\theta} = \omega \tag{2.3}$$

$$\omega = \omega^* + d_\omega(p^* - p) \tag{2.4}$$

Here, d_ω represents the droop gain. Equation (2.6) resembles equation (2.1). The synchronous machine model typically has an automatic voltage regulator (AVR). The AVR maintains the output voltage of the synchronous generator at a reference value independently of the speed and the load of the synchronous machine [12] [13] [14]. In the droop model it can be substituted by a PI regulator to compensate the output voltage error of the inverter, as shown in (2.5) with $\hat{v_d}$ being the direct axis reference, v^* and $||v||$ the reference and measured voltage magnitude.

$$\hat{v}_d = k_p(v^* - ||v||) + k_i \int_0^t (v^* - ||v(\tau)||)d\tau \tag{2.5}$$

Typically, to control three phase systems it is advantageous to change the reference frame and move from an *abc* reference frame to a *dq* reference frame, because the sinusoidal signals in the *abc* reference frame become dc signals in the *dq* reference frame. Therefore, it is easier to control the systems. The $abc \rightarrow dq$ and the inverse transformation will be explained later in Section 3.2.2.2. Referring to Figure 2.4, the instantaneous delivered power of the inverter is measured and compared to the power set-point. The error $\Delta p = d_\omega(p^* - p)$ is added to ω^* producing ω, which represents the frequency in rad\s. The integration of ω produces θ that represents the phase angle of the three phase system. $\hat{v_d}$ will be used as modulation signal to feed a PWM Generator block. In turn the PWM Generator block produces the PWM signals to control the inverter.

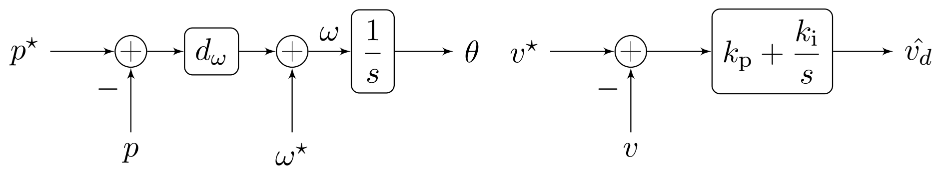


Figure 2.4: Droop control block diagram. On the left there is droop speed control characterized by the gain d_ω. On the right, the PI controller that gives the $\hat{v_d}$ reference voltage of the converter [10].

2.4 Virtual Synchronous Machine Control Technique

Another popular approach is the Virtual Synchronous Machine (VSM). The strategy is based on the idea of simulating the inertia of a synchronous machine. The synchronous machines, thanks to the stored kinetic energy, add rotational inertia that is a crucial property of frequency dynamics and stability. The VSM approach introduces the concept of "virtual" inertia. Since the system does not present a real physical inertia, the concept "virtual" inertia is most correlated to the concept of "emulating the behaviour of a system that has a real inertial", e.g., a synchronous machine. Therefore, the VSM control presents the same disadvantages of a synchronous machine, such as the loss of stability due to under-excitation and the oscillations around the synchronous frequency [5] [15] [16].

An advantage instead, is the possibility to tune the model parameters with more flexibility. In a synchronous machine the inertia, field inductance, mutual inductance, etc., are fixed by the physical structure of the synchronous generator. With the VSM approach instead, it is possible to tune the parameters because they do not derive from a physical characteristic of the generator. In addition, the disadvantages introduced by the non ideality of the synchronous machine, like the eddy currents and magnetic saturation that are strictly correlated to the physical property of the materials can be neglected [5][17].

Let's now introduce the VSM model, starting from the frequency dynamics described in (2.7).

$$\dot{\theta} = \omega \tag{2.6}$$

$$\ddot{\theta} = \frac{D_p}{J}(\omega^* - \omega) + \frac{1}{J\omega^*}(p^* - p) \tag{2.7}$$

Here, D_p is the virtual damping and J the inertia. Equation (2.7) reduces to (2.1) if $J/D_p \approx 0$. To show that, we rewrite (2.7) as shown in equation (2.8).

$$\omega^* - \omega = \frac{J}{D_p}\ddot{\theta} + d_\omega(p^* - p) \tag{2.8}$$

Where $d_\omega = \frac{D_p}{\omega^*}$. The phase angle of the three-phase system is given by integrating $\dot{\theta}$. θ is used to compute the three-phase voltage induced by the VSM. $\hat{v}_a$, $\hat{v}_b$, $\hat{v}_c$ are used as modulating signals to control the inverter.

$$\begin{bmatrix}\hat{v}_a\\ \hat{v}_b\\ \hat{v}_c\end{bmatrix} = 2\omega M_f i_f \begin{bmatrix}\sin\theta\\ \sin\left(\theta - \frac{2\pi}{3}\right)\\ \sin\left(\theta - \frac{4\pi}{3}\right)\end{bmatrix} \tag{2.9}$$

where M_f is the virtual mutual inductance and i_f is the excitation current. As before, in order to replicate the AVR of the synchronous machine, it is possible to regulate the excitation current with a PI controller as shown in (2.10).

$$i_f = \frac{k_p}{M_f}(v^* - ||v||) + \frac{k_i}{M_f}\int_0^t (v^* - ||v(\tau)||)d_\tau \tag{2.10}$$

According to equation (2.9), the injected current determines the three-phase voltage of the inverter. A block diagram of the VSM controller is shown in Figure 2.5.

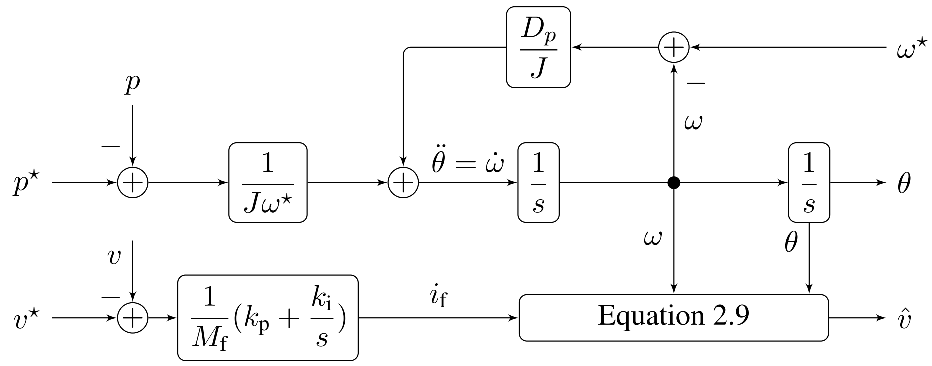


Figure 2.5: Virtual Synchronous Machine Controller [10].

2.5 Matching Control Technique

The Matching control is a novel control strategy introduced for the first time in [6]. The synchronous machine frequency is strictly correlated with the power, i.e. it can be used to indicate power imbalances. The dc link voltage of the inverter presents the same property, where power disturbances directly influence the dc link voltage of the converter [8] [18]. The objective of the grid forming control is to form the grid frequency and the grid voltage. Thanks to the previous observations, the dc voltage can be used to control the converter frequency as shown in (2.11).

$$\dot{\theta} = \omega = k_\omega v_{dc} \tag{2.11}$$

where $k_\omega = \omega^*/v_{dc}^*$. In order to control the ac voltage we use a PI controller that produces the modulation signal μ as shown in (2.12).

$$\mu = k_p(v^* - ||v||) + k_i \int_0^t (v^* - ||v(\tau)||)d\tau. \tag{2.12}$$

The reference voltage of the voltage controller in $\alpha\beta$-coordinate is given by:

$$\begin{bmatrix} \hat{v}_\alpha \\ \hat{v}_\beta \end{bmatrix} = \mu \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix} \tag{2.13}$$

that can be directly converted in the *abc*-coordinate system to produce the three modulation signals to control the inverter. A bock diagram of the Matching controller is shown in Figure 2.6.

To better highlight the similarity between the Matching controller and the synchronous machine let us introduce the two-level voltage source converter model shown in Figure 2.7 and described by equation (2.14). Here, C_{dc} is the dc-Link capacitance, G_{dc} the conductance that models the dc losses and L, R, C, the inductance, the resistance and the capacitance of the ac filter. i_{dc} is the

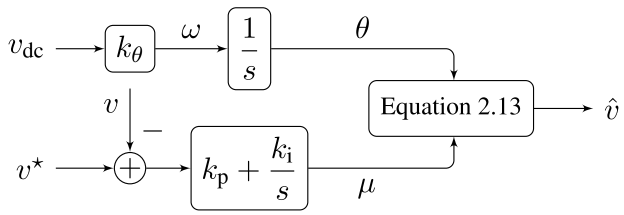


Figure 2.6: Matching Controller [10].

current injected by the controllable current source, the dc current provides to switching stage is represented by i_x and, i_s and i represent the ac current before and after the LC filter.

$$C_{dc}\dot{v}_{dc} = i_{dc} - G_{dc}v_{dc} - i_x, \tag{2.14a}$$
$$L\dot{i}_s = v_s - Ri_s - v, \tag{2.14b}$$
$$C\dot{v} = i_s - i \tag{2.14c}$$

Let us also introduce the second order synchronous machine model described by equation (2.15), where J_r is the total inertia of the rotor, D_ω is the damping coefficient, T_m the mechanical torque and T_e the electrical torque.

$$\dot{\theta} = \omega, \tag{2.15a}$$
$$J_r\dot{\omega} = T_m - D_\omega - T_e \tag{2.15b}$$

If we substitute v_{dc} with ω/k_θ in (2.14a) and then with reference to (2.11) we obtain the following second order system:

$$\dot{\theta} = \omega, \tag{2.16a}$$
$$C_{dc}\dot{\omega} = k_\theta i_{dc} - G_{dc}\omega - k_\theta i_x. \tag{2.16b}$$

Now the similarities between the synchronous machine and the inverter model are more clear. Equation (2.16b) and (2.15b) show how the dc-link capacitance mimics the rotational inertia of the synchronous machine. Referring to Figure 2.7, if the dc controller that injects i_{dc} is a proportional controller, we obtain the same behaviour that has the primary frequency control of the synchronous machine. Therefore matching control requires a P control to exhibit frequency droop behavior. However, if one is not interested in this property, matching control is more robust with dc side controlled via a PI controller.

2.6 Dispatchable Virtual Oscillator Control Technique

Dispatchable virtual oscillator control (dVOC) is a decentralized grid forming control strategy that has three main features. Firstly the user can specify the

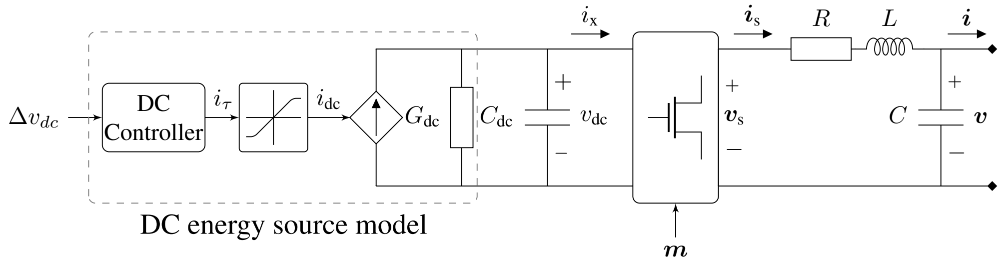


Figure 2.7: Two-level voltage source converter model

desired power set-point for each inverter. Secondly, if it has no set-point it behaves like a virtual oscillator control (VOC). Lastly, the dVOC control under the assumption that the set-points are consistent with the ac power flow makes the grid globally asymptotically stable [8] [7] [19] [20] [21].

The dVOC control low in the $\alpha\beta$-coordinates is given by equation (2.17), where $\hat{v} = [\hat{v_\alpha} \quad \hat{v_\beta}]^T$ is the reference voltage, while the injected current of the inverter is $i = [i_\alpha \quad i_\beta]^T$. The inductance to resistance ratio is controlled by the parameter $\kappa = \tan^{-1}(l\omega^*/r)$, while η, α are control gains.

$$\dot{\hat{v}} = \omega^* J_2 \hat{v} + \eta \left(K\hat{v} - R_2(\kappa)i + \frac{\alpha}{v^{*2}}(v{*}2 - ||\hat{v}^2||)\hat{v} \right) \tag{2.17}$$

Finally,

$$R_2(\kappa) := \begin{bmatrix} \cos\kappa & -\sin\kappa \\ \sin\kappa & \cos\kappa \end{bmatrix}, \quad K := \frac{1}{v^{*2}} R_2(\kappa) \begin{bmatrix} p^* & q^* \\ -q^* & p^* \end{bmatrix} \tag{2.18}$$

where R_2 is the 2-D rotation by κ.

If phase synchronization is achieved, i.e. $K\hat{v} - R_2(\kappa)i = 0$ and $(v^2 - ||\hat{v}^2||)\hat{v} = 0$, the dynamics in (2.17) reduce to an harmonic oscillator. To better highlight the droop characteristic of the dVOC control, let us rewrite equation (2.17) in polar coordinates considering an inductive network, i.e. $\kappa = \pi/2$, as shown in (2.19).

$$\dot{\theta} = \omega = \omega^* + \eta \left(\frac{p^*}{v^{*2}} - \frac{p}{||\hat{v}||^2} \right), \tag{2.19a}$$

$$||\dot{\hat{v}}|| = \eta \left(\frac{q^*}{v{*}2} - \frac{q}{||\hat{v}^2||} \right) ||\hat{v}|| + \frac{\eta\alpha}{v^{*2}} \left(v^{*2} - ||\hat{v}||^2 \right). \tag{2.19b}$$

If $v^* \approx ||\hat{v}||$, i.e. the network is near the nominal steady state, the dVOC control mimics the droop behaviour of (2.1) highlighting the relation between the frequency and the active power ($d_\omega = \eta/v^{*2}$). Equation (2.19b) reduces to the voltage regulator control $||\dot{\hat{v}}|| \approx -2\eta\alpha(||\hat{v}|| - v^*)$ choosing α such that the post-fault voltages are consistent with the other grid forming techniques because the first therm of equation (2.19b) becomes negligible [10] [22] [7]. A block diagram of the dVOC controller is shown in Figure 2.8.

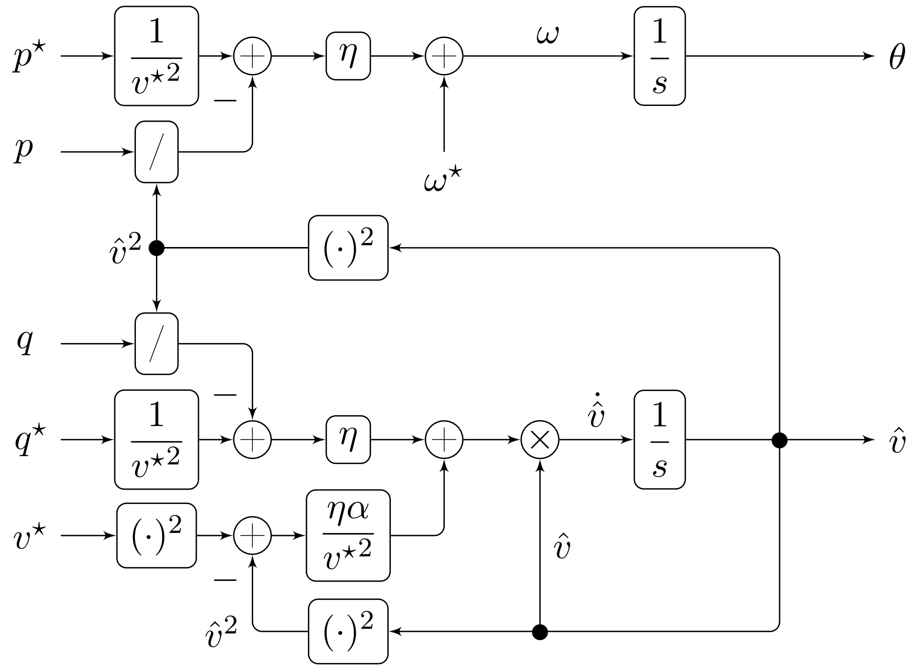


Figure 2.8: dVOC Controller

2.7 dc Voltage Control

To validate of the grid forming techniques presented above, we consider the basic model shown in Figure 2.7. To control the voltage v_{dc} we use a PI controller that regulates the injected current i_r. The PI controller automatically compensates the losses due to the conductance G_{dc} achieving zero steady error in the voltage regulation. The PI controller receives as input the error $\Delta v_{dc} = v^*_{dc} - v_{dc}$, where v^*_{dc} is the reference dc voltage and computes the current i_r to inject into the circuit.

For the VSM, dVOC and droop control the dc side can be controlled either by a proportional control or integral/PI control, while matching control requires a P control to achieve load sharing i.e. exhibiting frequency droop behavior. However, if one is not interested in this property, matching control is more robust with the dc side controlled via a PI controller, as described before. For this project we consider that all the techniques have a PI controller on the dc side. We are interested to perform control-hardware-in-the-loop simulations in order to test the stability of the controllers and not to compare their behaviour.

2.8 Conclusions

In this chapter we presented four novel grid forming techniques. We started explaining the importance of the synchronous generators and the concept of rotational inertia. Subsequently we described the primary frequency control of a synchronous machine. For each of the grid forming techniques we introduced the mathematical model presenting an analogy with the primary frequency control of the synchronous generator. All the novel techniques try to mimic the behaviour of the synchronous machine in order to address the challenge of the low inertia grid and to guarantee grid stability. The chapter ends with the description of the dc controller in order to provide a complete basic setup that will be used in the next chapters to perform real-time simulations.

Chapter 3

Components level validation

This chapter is focused on component level validation. The purpose is to validate the grid forming techniques presented in the previous chapter . In order to achieve this, we adopt the approach of Controller Hardware in the Loop (CHIL) simulations. Four different controllers (droop, VSM, matching, dVOC) will be tested and validated. The chapter starts with an overview of the CHIL validation setup, then it presents the methodology used to configure the hardware and run the simulations. Several test cases were performed to validate the control strategies under investigation.

3.1 Introduction

The renewable energy is increasing considerably in recent years, by promoting the development of new control techniques. In particular in the field of research many studies about grid forming strategies have been done. Nowadays, the majority of the tests made to prove the stability and reliability of these techniques are just offline simulations and no tests involving real hardware have been performed so far. However, an offline simulation does not seam to be sufficient to proof the stability of the techniques. Therefore, it would be advisable to use different tools that also test real hardware. Figure 3.1 shows a diagram of the complexity\cost vs. validation accuracy of the various validation techniques. Starting from the low level an offline simulation is the least complex and also the least accurate validation approach. The dynamic models of the system under test and the controllers are simplified and implemented in a simulation software environment (e.g. Simulink). A PC hosts the simulation software and no real hardware is tested.

The next level is the CHIL approach. In literature it is common to find also the terms Controller in the Loop (CIL) and Hardware in the Loop (HIL), usually they are strictly related and for that reason we consider them as CHIL. A CHIL simulation is an approach used to test the controller. The dynamic model of the system is implemented inside a real-time simulator (e.g. PLECS RT Box) and the controller inside a microcontroller (e.g. TI LaunchPad F28379D). Typically the microcontroller is connected to the simulator via ADC and DAC converters or digital I\Os. This scenario compared to the previous one requires a RT-simulator that usually is expensive, but has several advantages. First of all, the accuracy increases because we have real hardware under test. The behaviour of the controller can be analyzed without connecting it to the power circuit/grid and thus avoiding the possibility of destroying components.

The next level is the Power Hardware in the Loop simulations where the hardware under test is a power electronic device (inverters, batteries, etc). While in CHIL the simulator and the microcontroller exchange low level signals, now the signals at stake are power signals. For that reason power amplifiers and sensors are required to amplify and measures the signals from and to the controller. From a control point of view an amplifier can introduce instability because it introduces new dynamics in the system. The validation accuracy increase because the PHIL approach guarantees that the controller works fine with the hardware under test that in a future will be the hardware used in a laboratory prototype. As expected, the complexity and the cost of the system increases due to the power amplifier and the real hardware.

The last level is the field validation. A complete prototype of the project is built in the laboratory. In this case the prototype is fully functional an integrates the controller and the power hardware tested in the previous simulations. In terms of cost it is the most expensive, but also the most accurate because it represents accurately a complete physical product without any simplifications.

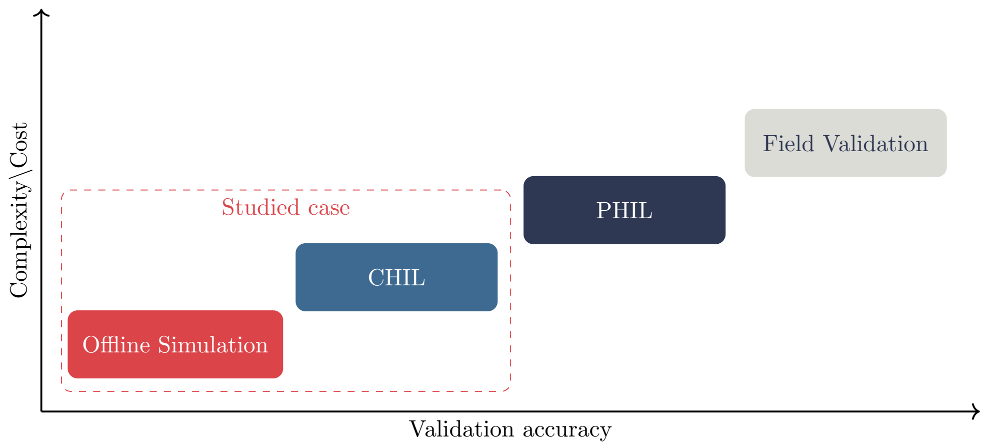


Figure 3.1: Complexity\cost vs. validation accuracy of the various types of validation approaches.

In this chapter we perform component level validation, i.e. we test and validate a single components, for example a control card that runs controllers based on grid forming techniques. We are interested to test these controllers in real-time, trying to understand if they are stable. All the aspects regarding technical challenges such as the signals transmission, the hardware synchronization and the hardware limits are taken into account. The attention is restricted to offline simulations and CHIL simulations, as highlighted with the red rectangle in Figure 3.1. Subsequently, in the next chapter we proceed towards system level validation which compared to the component level validation, performs simulations of multiple controllers connected to a larger grid.

3.2 Methodology

Before analyzing the CHIL validation setup it is useful to take a short glance at the main differences between an offline simulation and a CHIL simulation. Figure 3.2 presents from a high point of view the outline of an offline simulation and a CHIL simulation. The starting point is always an offline simulation. The physical system and the controller are modelled inside a simulation software (e.g. Simulink, PLECS) hosted on a standard PC. The signals exchanged between the plant and the controller model are numerical signals inside the simulations and no real hardware is involved. Since the beginning, as marked in the diagram, it is important to distinguish the plant from the controller model because in the next steps they will be separated and deployed on different hardware devices.

The second diagram of Figure 3.2 outlines a CHIL setup. It is composed of two main parts: on the left we have the real-time simulator which emulates the physical system by running its mathematical model in real-time. On the right, the control card which runs control algorithm is shown. The software of the real-time simulator usually provides a function to automatically generate c-code from the plant model. By compiling this code a target file is produced that is then loaded inside the real-time simulator. This setup is usually called HIL because the hardware under test runs in real-time inside the simulator and can be use to test the controller. Unfortunately the controller model cannot be automatically programwed inside the control card and the user has to manually program it as c-code. The conversion of the controller model in c-code is one of the challenges addressed in this chapter and one of the main contributions of this project. The controller runs in real time inside the microcontroller and for that reason the configuration is named Controller in the loop. Physical cables connect the control card to the real-time simulator that emulates the physical system. The Real-time simulator provides measurements of the physical system in form of analog signals. The control card that runs the controller reads the measurements and sends back control signals (e.g. PWM) to the real-time simulator, thus close the loop.

In the next paragraphs we present the methodology used to run a CHIL simulation, showing in details all the required configuration and limitations of the systems and the hardware under test.

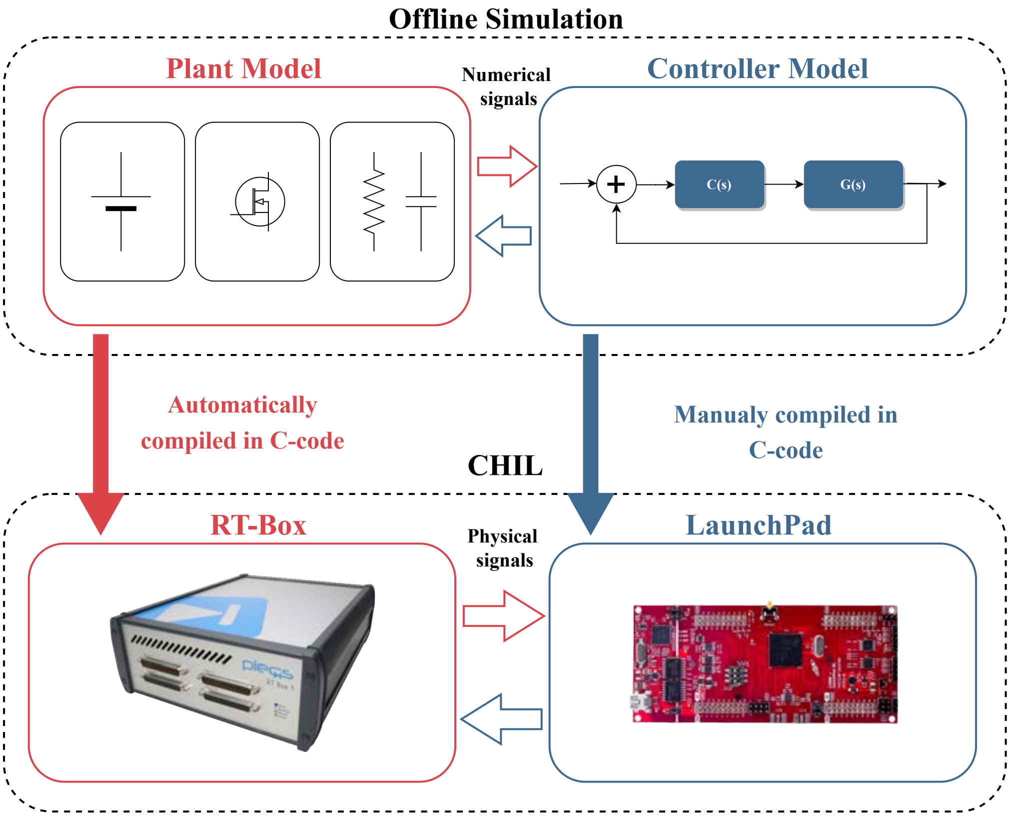


Figure 3.2: Conceptual representation of the offline and CHIL simulations

3.2.1 CHIL Validation Setup

In order to validate the grid forming techniques presented in the previous chapter we employed a controller hardware in the loop (CHIL) approach. In broad terms, the used setup was composed of the following main components:

- a real-time simulator, more specifically, the Plexim RT Box,
- an industrial grade microcontroller development kit: C2000 Delfino MCU F28379D LaunchPad,
- an interface board that connects the microcontroller to the real-time simulator: RT Box LaunchPad Interface.
- a PC that connects to both the microcontroller and the real-time simulator and hosts all the required software.

Figure 3.3: CIL Validation Setup composed of Plexim RT Box real-time simulator and F28379D microcontroller

In terms of interfacing options, the real-time simulator, i.e. the RT Box, has four DB37 connectors. Two for digital inputs and outputs (IOs) and another two for analog IOs. [23]. The interface board connects directly to the RT Box through these connectors and is equipped with the necessary sockets required to host the LaunchPad development kit. In this way, the interface board acts like a bridge that connects the controller and the simulator. Moreover, the interface board provides the power supply to the F28379D and has several test points which can be used for directly connecting an oscilloscope to monitor the signals.An overview of the hardware used in the setup is shown in Figure 3.3.

From the software perspective, the F28379D microcontroller is equipped with a module named XDS100v2-On-Board Debug Probe that enables JTAG debugging/programming as well as serial communication over USB to the host PC [24]. Therefore, in order to program and debug the microcontroller an USB connection to the host PC is required. The software used for programming the microcontroller is Code Composer Studio v8, a development suite provided by the microcontroller's manufacturer, i.e. Texas Instruments.

The real-time simulator connects to the host PC via Ethernet. Models for the simulator are developed and compiled using a software package specialized for modeling power electronics systems named PLECS.

3.2.2 Plexim RT Box

Plexim RT Box is a real-time simulator used for HIL or CIL testing, but also for fast control prototyping [25]. The computation core of the simulator is a Xilinx Zynq system-on-chip. Figure 3.4 presents the architecture of the RT Box. Core 1 of the on-board ARM processor is in charge of the user interface via the Ethernet connection. Meanwhile, core 2 runs the real-time application and it is interfaced with the FPGA which provides the digital and analog IO channels.

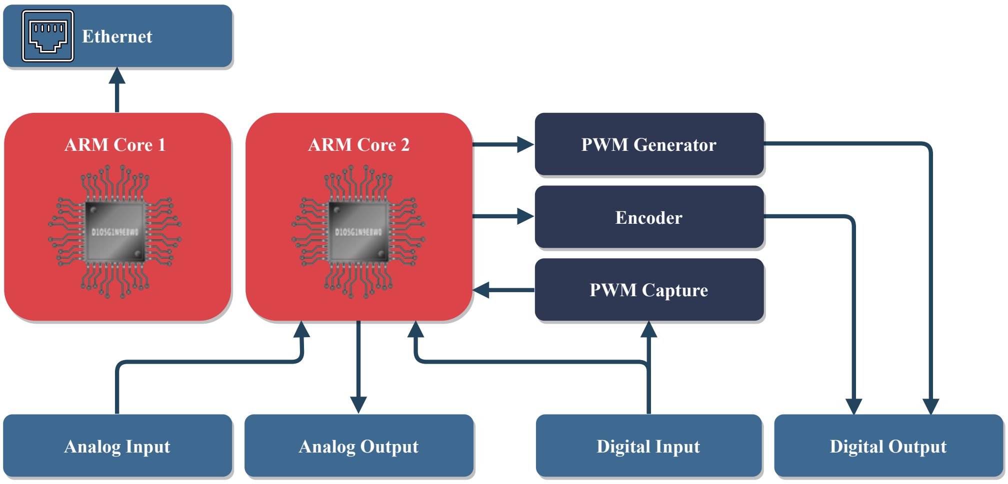


Figure 3.4: Architecture of the RT Box

Since the RT Box works in real-time, the user can choose the desire sampling frequency f_s or equivalently the sample time $T_s = \frac{1}{f_s}$ of the model, but it has to satisfy the hard time criteria. This criteria imposes that the execution time T_{load} of the simulator is smaller then T_s. T_{load} is the time required to read and process the input values, but also to calculate a model step and to write the outputs to the DAC converters. A graphical representation of the criteria is shown in Figure 3.5.

The RT Box allows to monitor the executing time (T_{load}) of the CPU in the monitor windows. If $T_{\text{load}} < T_s$ for a certain period T_{idle}, the CPU is in *idle* mode, which means that it is not computing anything. The percentage usage of the CPU is calculated with the following formula: $CPU_{\text{load}} = \frac{T_{\text{load}}}{T_s}$. If $T_{\text{load}} > T_s$ the CPU goes in overrun mode and the simulator presents an error state. In this case the sample time is too small and the user has to increase it or decrease the complexity of the model. All the digital and analog signals are updated once every sample period T_s. However, in particular occasions high

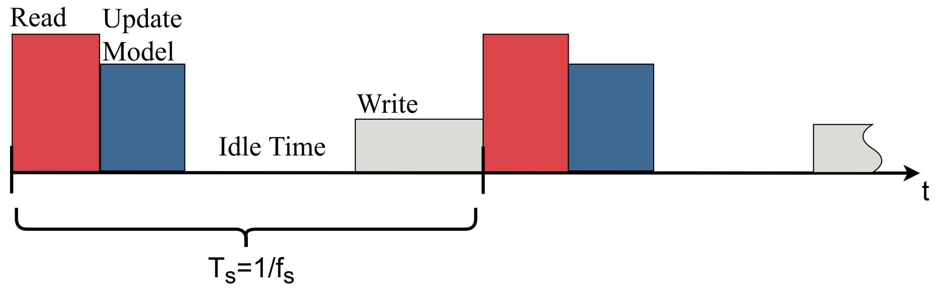


Figure 3.5: Hard real-time criteria.

frequency input signals need to be captured and the sample frequency f_s is not fast enough. Fortunately, the RT Box provides an efficient module called PWM-Capture. This module captures the input signals with a resolution of 10ns providing as output a value that represents the average of the time in which the signal was high and low during the period T_s. The PWM-Capture module plays a fundamental role when the user works with PWM signals which usually have high frequency with variable duty cycle. Considering all the elements of the RT Box presented before, the simulator has proven a certain level of versatility and flexibility. Therefore, it seems suitable for a wide range of applications. For that reason, there are some aspects that need to be clarified in order to understand the capability of the system and the proper way to use it. The simulator can be used for two types of applications, mainly:

- HIL simulations/CIL validation, and
- Rapid Control Prototyping.

In the first mode the simulator emulates the power stage of a power electronics system. For example, in our case it emulates a two-level three phase inverter together with its dc side. The control card and the control algorithms typically referred to as the "controller", are usually the devices under test. In our case, the control card is the F28379D and the algorithms are the grid-forming control strategies.

In a real prototype the inputs to the controller would be analog signals coming from the sensors mounted in the physical system. The RT Box emulates these sensors and provides the required signals via DACs with a 16 bit resolution at a maximum sample rate of 2 Msps [25]. On the other side, the controller produces PWM signals to control the power electronics switches. The RT Box through the PWM capture module offers an interface with a latency of a few microseconds. In this way the controller cannot distinguish weather it is connected to a simulator or to a real inverter. Therefore, connecting the controller to the simulator allows us to perform exhaustive tests without the need of the physical power stage and a high voltage laboratory.

The other approach is to use the RT Box for rapid control prototyping. In this sort of applications the simulator acts like a controller and it provides the necessary signals to control power electronic components, e.g., a real inverter or another RT Box that emulates the power stage of an inverter.

This type of set-up has several important advantages that can be summarized in:

- Flexibility to change the electrical model in accordance with various use-case requirements,
- User and equipment safety. Since all the power components are emulated, there is no risk if something goes wrong during the control development stage,
- Scalability. Multiple RT Boxes can be connected together in order to simulate scenarios that involve a larger number of components.

Figure 3.6: Plexim RT Box [25]

3.2.2.1 Modelling Environment

The RT Box comes with a complete software environment called PLECS. PLECS is an autonomous software package for time-domain simulation of power electronic systems [26]. Before being a standalone application, PLECS was initially designed, and it is still being used, as a Simulink toolbox. Therefore, for users that have previously worked with Simulink, working with PLECS will feel rather familiar. The majority of the frequently used Simulink blocks are also implemented in PLECS and usually have the same parameters. Similar to Simulink, the software provides basic building blocks that enable the user to build more advanced functionality. Application specific blocks, such as, for example in our case, a PLL or specific controllers, had to be custom built. In this way, the user can create a personal library to be used in different applications.

3.2.2.2 PLECS C-scripts library

PLECS provides a block called C-scripts where it is possible to implement any desire function in c-code. Since the PLECS library doesn't provide all the necessary blocks, we developed a custom one. In this way the migration from the offline simulation to CIL is faster because most of the logic it is already written as code snippets. Our custom library provides the following blocks:

Clarke transformations: The Clarke transformation ($abc \rightarrow \alpha\beta$) is widely used in the field of three phase systems because allows the representation of a three phase signal into a two dimensional vector. The transformation matrix is shown in (3.1) and and the inverse transformation is shown in (3.2)

$$\begin{bmatrix} y_\alpha \\ y_\beta \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \tag{3.1}$$

$$\begin{bmatrix} y_a \\ y_b \\ y_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} \tag{3.2}$$

The Clarke transformation is used in the grid forming techniques, for example in the matching control, as shown in the previous chapter.

Park transformations: ($abc \rightarrow dq$) in the context of the three phase systems the park transformation is equally wide spread. It consists of a signal transformation from the *abc* reference system to a rotating frame system. The input is a three phase signal in the *abc* stationary reference frame and the outputs are a direct and quadrature signals in a rationing reference system. The park transformation matrix is shown in (3.3) and the inverse transformation in (3.4)

$$\begin{bmatrix} y_d \\ y_q \end{bmatrix} = \frac{2}{3} \begin{bmatrix} \cos\theta & -\sin\theta \\ \cos(\theta - \frac{2\pi}{3}) & -\sin(\theta - \frac{2\pi}{3}) \\ \cos(\theta + \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) \end{bmatrix}^T \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \tag{3.3}$$

$$\begin{bmatrix} y_a \\ y_b \\ y_c \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \cos(\theta - \frac{2\pi}{3}) & -\sin(\theta - \frac{2\pi}{3}) \\ \cos(\theta + \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) \end{bmatrix} \begin{bmatrix} x_d \\ x_q \end{bmatrix} \tag{3.4}$$

The Park transformation is very useful because it simplifies a lot the analysis of three phase systems. The ac signals are transformed to dc signals in the *dq* reference frame. Hence, PID controllers can be used instead of more expensive and complicated controllers, ensuring a zero error tracking. A disadvantage of using the dq frame is that the system requires a PLL block to extract the phase angle θ from the V_{abc} voltages, as Θ is used to compute the transformation and synchronize the voltages of the converter to those of the grid.

Virtual oscillator controller is a block that receives as input the frequency of the system $\omega[rad/s]$ and returns the phase θ. The principal component of the block is a discrete integrator. The block bounds the output between 0 and 2π. If the output reaches 2π, the internal state of the integrator is reset to zero. The VOC is implemented with Algorithm 1. The VCO is used because the three phase systems works with periodical signals of period 2π. In practice a discrete integrator cannot be used because the output continues to grow leading to an overflow error of the variable that stores it.

Algorithm 1 VOC

Input: ω
Output: θ
Initialization: Previous integrator state $S_{t-1} \leftarrow 0$;
$\theta \leftarrow S_{t-1} + \omega T_s$;
if $\theta > 2\pi$ **then**
  $\theta \leftarrow 0$;
end if
if $\theta < 0$ **then**
  $\theta \leftarrow 2\pi$;
end if
$S_{t-1} \leftarrow \theta$;
Return: θ;

PLL-Phase Lock Loop: is used in the grid following simulations. The output of the PLL is a signal whose phase is correlated with the input signal. It is used in the grid following scenario where the output voltage of the converter must be in phase with the grid voltage vector. Figure 3.7 shows a functional diagram of a PLL. Since for the converter the phase angle of the grid is unknown, the idea is to extract this information from the measurement of the three phase voltages V_{abc}. An $abc \to dq$ block transforms the input voltages in the synchronous reference frame dq. Let's define U_d^* as the reference voltage of the d-axis and U_d as the current voltage of the d-axis of the grid computed with the $abc \to dq$ block. To reach the "in phase" condition the $\Delta U_d = U_d^* - U_d$ must be zero. ΔU_d is the error, i.e. the misalignment between the grid voltage vector and the output voltage of the converter, a PI controller compensate the error as long as $\Delta U_d = 0$. The PI controller output added to the reference frequency w_{ff}(usually 50Hz or 60Hz) provides the grid frequency. w feeds a Virtual controller oscillator (VCO). The output of the VCO is the phase Θ of the three phases system [27].

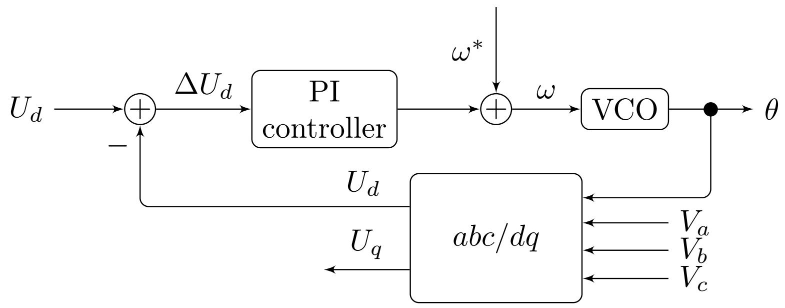


Figure 3.7: Phase Lock Loop functional diagram

PID controller The PID (Proportional Integrate Derivative) controller is one of the most used control strategy in practice because it is simple to implement and tune. The equation of the PID controller is shown in (3.5) and in the Laplace domain in (3.6).

$$f(t) = K_p e(t) + K_i \int e(\tau) d\tau + K_d \frac{de(t)}{dt} \tag{3.5}$$

$$F(s) = K_p E(s) + \frac{K_i}{s} E(s) + K_d s E(s) \tag{3.6}$$

The PID controller is composed of three main blocks, as shown in Figure 3.8:

- Proportional Block: is composed of a gain K_p, the output is proportional to the error according to K_p. Usually, the action of the proportional therm is not sufficient to control a system because the system may present oscillations and a poor reference tracking.

- Integral Block: computes the integral of the error. This means that it evaluates not only the error, but also the time for which it has persisted. The integral terms reduces the steady state error to zero.

- Derivative Block: computes the derivative of the error. It gives stability to the system reducing the overshoot and the oscillations.

To implement the PID controller in c-code the discrete version shown in (3.7) is considered.

$$F(z) = K_p E(z) + K_i \frac{T_s}{z-1} E(z) + K_d \frac{1}{Ts} \frac{z-1}{z} E(s) \tag{3.7}$$

where T_s is the sample time of the controller. The Algorithm 2 represents a possible implementation of the PID controller.

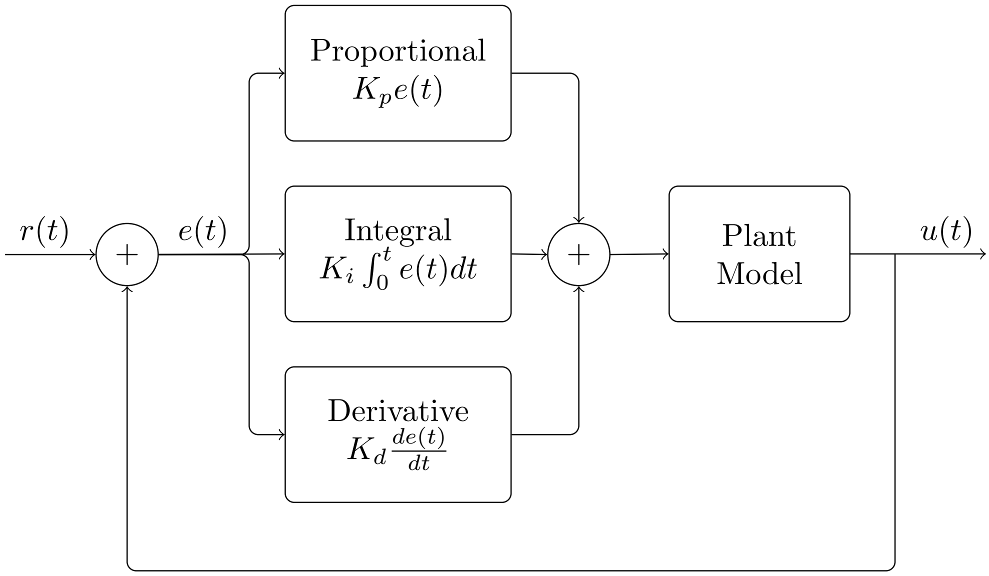


Figure 3.8: PID block diagram

Algorithm 2 PID

Input: Error $e_t \leftarrow$ Setpoint - MeasuredValue;
Output: U_t;
Initialization: Previous integrator state $i_{t-1} \leftarrow 0$;
Previous error $e_{t-1} \leftarrow 0$;
$i_t \leftarrow i_{t-1} + e_t T_s$;
$d_t \leftarrow (e_t - e_{t-1})/T_s$;
$U_t \leftarrow K_p e_t + K_i i_t + K_d d_t$;
$e_{t-1} \leftarrow e_t$;
$i_{t-1} \leftarrow i_t$;
Return: U_t

Power measurement block The power measurement block measures the instant power in per unit. It has as input the voltages V_{abc} and currents I_{abc} that are transformed in per unit dividing them by the nominal voltage and nominal current respectively. The Park transformation is applied to V_{abc} and I_{abc} to obtain V_d V_q and I_d I_q. At this point the equation (3.8) computes the instant power.

$$\begin{aligned} P &= \frac{3}{2}(V_d I_d + V_q I_q) \\ Q &= \frac{3}{2}(V_q I_d - V_d I_q); \end{aligned} \tag{3.8}$$

It is a good practice to add a first order low pass filter because usually the measurements are noisy.

Magnitude measurement block This block computes the instant magnitude of the three phase input signal. The input is converted to the *dq* reference frame and after that equation (3.9) is used to compute the magnitude.

$$r_{mag} = \sqrt{x_q^2 + x_q^2} \tag{3.9}$$

The voltage and current magnitude is used in the grid forming techniques not only for internal computation, but also for monitoring the behaviour of the controllers.

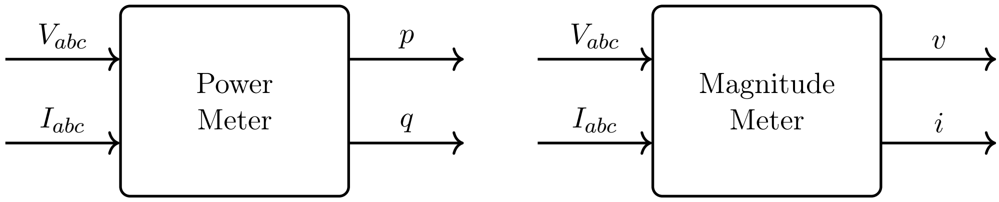


Figure 3.9: Meter blocks

3.2.3 Component level validation strategy

The approach that we used for component level validation involves multiple systems. Each of them has to be adequately configured in order to match the project specifications and ensure there is compatibility with the other parts of the setup. Due to the complexity of the system it is useful follow to a guideline step by step as represented below:

Offline simulation: Hardware Design The starting point is always an offline simulation. For simplicity the model is split in two parts: the power hardware and the controller. This will be useful subsequently when we move from the offline simulation to the CHIL simulation. For this project the power components are presented in Figure 3.10. On the left side the current control source, the resistor and the capacitor compose the dc circuit. The controlled current source injects current that charges the capacitor. The voltage across the capacitor feeds the two level three phases inverter. Six PWM signals control the MOSFETs gates of the inverter. The output of the inverter are three high frequency square waves with a variable duty cycle. To obtain three sine waves we have to filter them using a low pass filter. In our case that filter is an LC filter with a cut off frequency of 159Hz. On the right side of the diagram the rectangle box represents a generic grid. More than one tests will be executed and for each of them the model of the grid will be changed, in order to test the setup with different configurations. In particular, the inverter will be connected

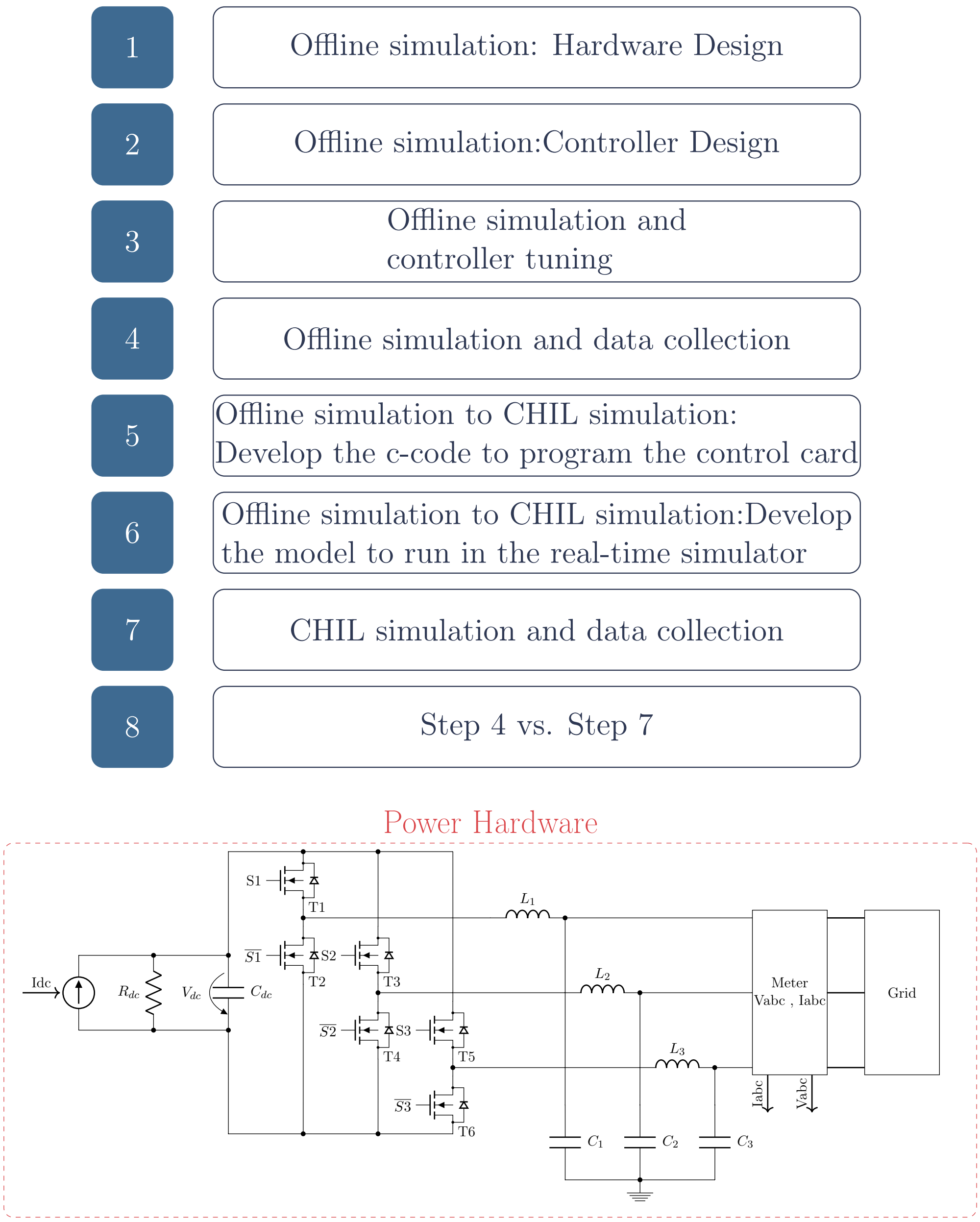


Figure 3.10: Hardware in the loop

to an infinite bus as well as to ZIP loads (constant impedance, constant current an constant power load respectively).

Offline simulation: Controller Design The second step requires the development of the controllers. Two controllers are modelled in order to operate the inverters, one for the dc side and one for the ac side. As the focus of the thesis is on the ac side, the controller on the dc side is kept the same for all the scenarios that we tested. It computes how much current should be injected in the system in order to keep the capacitor voltage at the reference level. It receives as inputs the reference voltage and the measured capacitor voltage (Vdc). As output it produces the dc current reference to inject inside the circuit through the controlled current source. The grid forming controller instead, implements the grid forming techniques (Matching, Droop, dVOC, VSM). Figure 3.11 shows a representation of the offline simulation model. The controller receives as inputs V_{dc}, I_{abc}, V_{abc}, i.e. the dc voltage, the line to neutral current for each phase, and the line to neutral voltage for each phase, respectively. The measurements come from the Meter block that is positioned just after the filter. The controller uses the measurements to compute the three modulation signals ,one for each phase, and the frequency of the converter. The PWM Generator block receives the modulation signals as inputs and produces as output three PWM signals to control the inverter. The inverter in Figure 3.11 has three legs composed by the MOSFTEs T1&T2, T3&T4 and T5&T6, respectively. To obtain the signals that control the three lower MOSFTEs of each leg one solution is to negate the PWM signals used to control the upper MOSFETs. To avoid shortcircuits, the MOSFETs of the same lag cannot be active at the same time because these devices required few nano-seconds to turning on or off. Therefore, in a real inverter it is necessary to add a dead-time between the signals that control the MOSFETs in the same leg.

Offline simulation and controller tuning: The next step consists of running the simulation and tuning the controller. This part can takes a lot of time because fine tuning is not a straightforward process. Anyway, this is an important step because it defines exactly all the parameters used in the real model. Having a fine tuned controller simplifies a lot the work with the real system and helps debugging the code. We tune the parameters of the controllers that implement the matching, dVOC, VSM and droop strategy based on the parameters provides in Table 1 of [8]. The tuning criteria adopted makes sure that all the controllers exhibit an identical frequency droop behaviour in presence of a load change. Since the parameters in Table 1 of [8] do not perform well making our model in-stable we tuned again all the PI controllers using the Ziegler-Nichols approach, while for the other parameters we followed a trial and error approach.

The Ziegler-Nichols approach is based on three main steps that can be summarize as follow:

- Set the integral term to zero and perturb the system with a load step. Increase the proportional gain until the output of the control loop presents stable and consistent oscillations, note this gain with K_u.
- Set $K_p = 0.45K_u$.
- Set $Ti = \frac{K_p}{K_i} = \frac{T_0}{1.2}$, where T_0 is the period of the oscillations.

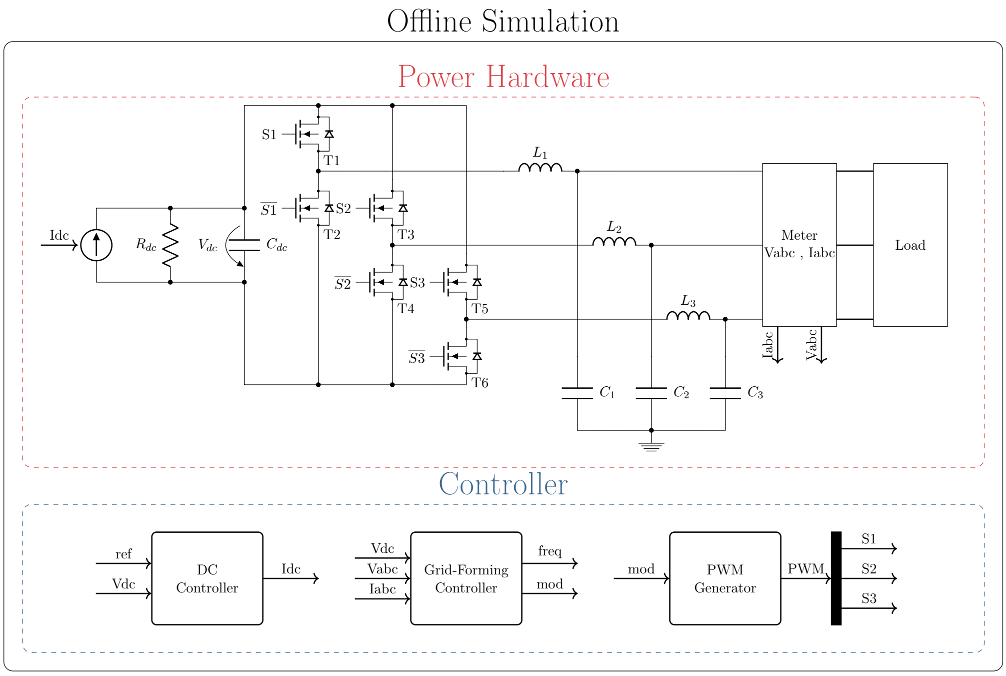


Figure 3.11: Offline simulation diagram

The case study parameters are reported in Table 3.1.

dc voltage control	
k_{dc_p}	8.4
k_{dc_i}	20.0

droop control		**VSM**	
d_ω	$2\pi 0.5$	D_p	10^5
ω^*	$2\pi 50$	J	2×10^3
k_p, k_i	1.0, 0.8	k_p, k_i	0.001, 0.03

matching control		**dVOC**	
k_{dc_p}, k_{dc_i}	8.4, 20.0	η	1.3
k_Θ	0.12	α	0.1
k_p, k_i	2, 30	κ	$\frac{\pi}{2}$

Table 3.1: Case study parameters

Offline simulation & data collection Performing the offline simulation and the data collection allows to verify if the controller is properly tuned and if the model presents some problems. It is recommendable to correct all the problems at this stage to avoid to modify the CHIL simulation that in general requires more time to apply a change.

Offline simulation to CHIL simulation: Develop the model to run in the real-time simulator Until now we used the PLECS software just for offline simulations. The next step consist of developing the model to run in the real-time simulator. The model is split in two parts and modified to obtain a model for the RT Box and the implementation of the controller in c-code inside the control board. The model that runs in the real-time simulator is shown in Figure 3.12. It is the same as the one presented in the Figure 3.11 with the exception that now the IO-interface substitutes the controller. In the next sections the configuration of the IO-interface will be explained in details.

Offline simulation to CHIL simulation: Develop the c-code to program the control card: The controller is translated in c-code and loaded inside the LaunchPad. Having the simulation model properly tuned and composed by the custom c-script blocks previously described reduce a lot the effort required by this step.

CHIL simulation & data collection: at this point it is possible to run the CHIL simulation and collect the data. The PLECS oscilloscope feature is very useful because it allows to monitor what is going on in real time and understand immediately if something does not work correctly. We are interest to monitor the network frequency, the voltage and current magnitude and the instant power of the converter to verify the stability of the converters. Unfortunately, not all these in formations can be capture from the RT Box because some of them are variables that reside inside the control card. Therefore, to overcame this problem we developed a strategy where the data is collected with the control card and then it is sent via a serial port to a PC.

Compare the results: Step 4 vs. Step 7 the last step is to compare the offline simulation results with the CHIL simulation results. In the comparison we take in exam a lot of factors, for example if and how the signals delays influence the system, how close the offline simulation results are to the real system results and so on. Everything will be describe in depth subsequently.

3.2.4 Hardware communication

This section explains how all the signals between the controller and the simulator are mapped and configured.

The interface board has four sockets to directly plug the LaunchPad and three DB37 connectors: one for Digital Inputs, one for Digital Outputs and one for Analog Outputs. These connectors are used to mechanically electrically connect the interface board with the RT Box. Unfortunately, an analog input interface to the RT Box is not present. This can be a limit for the applications that requires to send an analog signal to the RT Box.

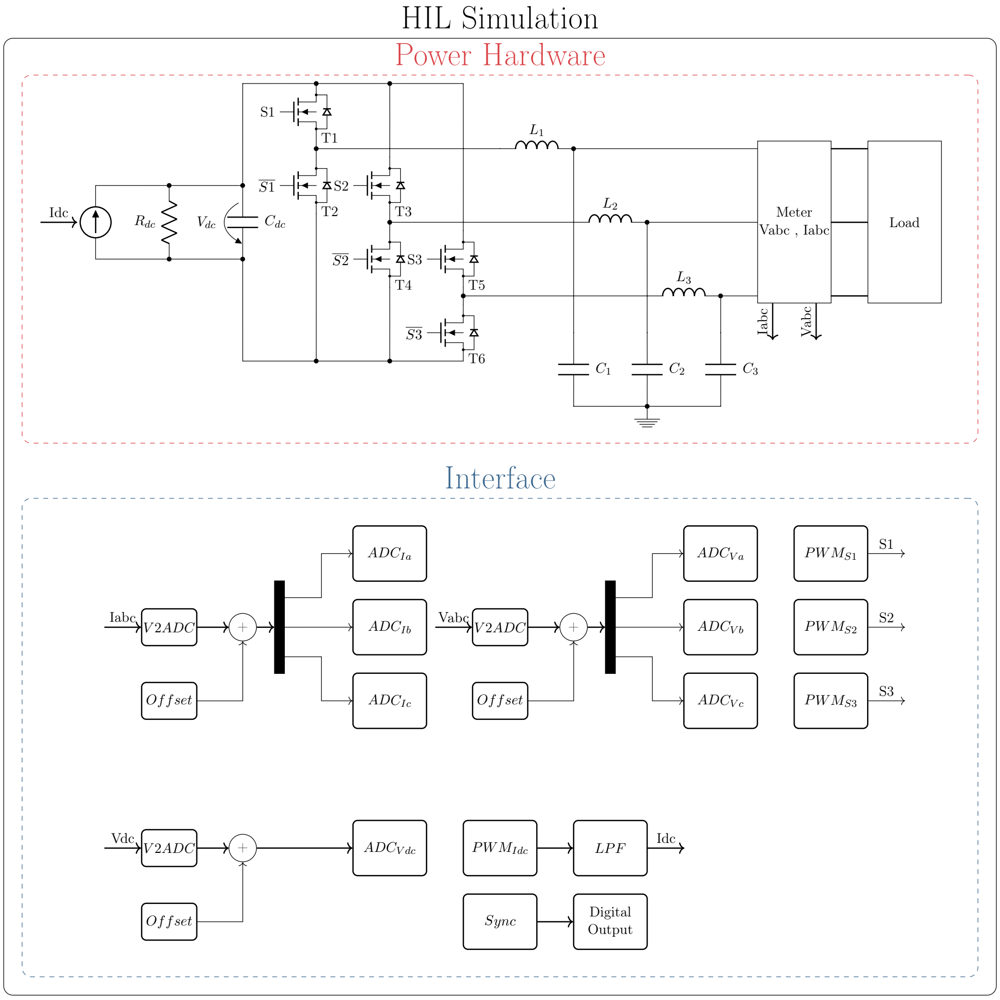


Figure 3.12: System model and controller interface running in the real-time simulator

To run the CHIL simulation the LaunchPad and the RT Box exchange signals. In particular, it is required to read V_{abc}, I_{abc} and V_{dc} from the RT Box and to send the PWM_{abc} signals to the real-time simulator to control the MOSFET gates. The PWMs are digital signals, instead ,V_{abc}, I_{abc} and V_{dc} are analog signals. The complete signal map of them is shown in Table 3.2.

An important aspect to consider is the electrical compatibility of the signals between the simulator and the controller. Let us briefly introduce the RT Box and LaunchPad specifications and a way to adequately configure the parameters (all the signal types, i.e. inputs or outputs, are referred which respect to the simulator).

RT Box	Pin Header	Channel	Signal
AO7	30	ADCINA0	Va
AO2	25	ADCINB3	Vb
AO1	24	ADCINC3	Vc
AO3	26	ADCINA3	Ia
AO5	28	ADCINB2	Ib
AO4	27	ADCINC2	Ic
AO11	66	ADCINA5	Vdc
DI16	80	PWM4A	PWM_a
DI4	36	PWM3A	PWM_b
DI8	78	PWM5A	PWM_c
DI5	35	PWM3B	PWM_{Idc}

Table 3.2: Signals configuration

Digital Input: For the digital inputs the voltage level parameter has to be set in PLECS. This is achieved by following the following menu path:
Coder->Coder Options->Target->Digital output voltage level->3.3V

The LaunchPad has a 0-3.3V voltage range for the outputs. Therefore, the inputs of the simulator must be set accordingly to this range. For receiving the signals in PLCES there are two different blocks, i.e.:

- **PWM Capture:** the output of the block represents the percentage of time during which a digital input signal was active over the last model step period [25].
- **Digital Input**: the output signal is 1 if the input voltage is higher than 2 Volts and 0 if it is lower than 0.8 Volts. For other input voltages the output signal is undefined. [25].

The PWM Capture block is developed specifically for capturing high frequency PWM signals so it is a good choice for this project, as the control card sends high frequency PWM signals to the RT Box.

Digital Output: PLECS provides a Digital Output block used to send digital signals to the control card. The control card can receive digital input signals in the range of 0-3.3V. Therefore, in the setting parameters of the Digital Output block it is important to set the output voltage level to 3.3V.

Analog Output: the range of the analog output signals can be set by the user. It is possible to choose four different ranges (0..5V, 0..10V, -5..5V, -10..10V) but none of them is compatible with the control card range that is

(0..3V). Unfortunately, the interface does not provide a dedicated circuit to convert the signals. Therefore, it is necessary to find a solution via software. The RT Box range closest to the controller is 0..5V. That can be set in the PLECS environment at the following menu path:

Coder->Coder Options->Target->Analog Output Voltage range->0..5V

Subsequently configure the Analog Out block by choosing the channel and set as maximum output voltage 3V. In this way if the input signal to the Analog block exceeds the limit the signal is cut. In other words the Analog Block presents a saturation block. A software solution is not the best because if the user forgets to set the correct parameters it might destroy the controller.

A hardware solution would be better because it could ensure that the signal does not exceed the limits. Unfortunately, with this solution it is not possible to represent negative values. Imagine for example a sine wave that for the half of the period is positive and for the other negative; the system accepts only positive voltage values. The signals must be modified to match the specifications: The process is done in several steps:

1. Convert the signal in per unit; in this way it is possible to change the system specifications as the voltage level and the nominal power without re-tuning the controllers. The user has only to adjust the conversion gains and not tune again the controller. The controller works in per unit and not in SI for the same reason.

2. Map the signal in the desired range.

3. Multiply the signal with an appropriate gain to correct when the signal exceeds the nominal value.

4. Add an offset to make the signal positive.

Let's clarify the procedure with an example: Consider a sine wave with an amplitude of 2V, which is the nominal value. To transmit it to the controller, firstly it is necessary to convert it in per unit by dividing by 2. The resulting signal has an amplitude of 1V, with a peak to peak of 2V. In the second step the signal is maped inside the desired interval. The range of the sine wave is -1..+1V so to map it inside the range -1.5..1.5V (having a peak to peak of 3V) we multiply it by 3/2. In the last step an offset of 1.5V is added to the sine wave to make it positive. Sometimes it is convenient to suppose that the signal will exceed the limits. Imagine that our sine wave at a certain time reaches the amplitude of 3.63V, so 110% of the nominal value. The Analog block cuts the input signal because it exceeds the limits. Executing the conversion, the signal goes from -0.15..3.15V. A simple gain before the offset solves the problem. The user can set it according to the application. In this case a good choice can be a gain of 1/1.1 that compresses the signal to the range of 0..3V. Figure 3.13 shows the process.

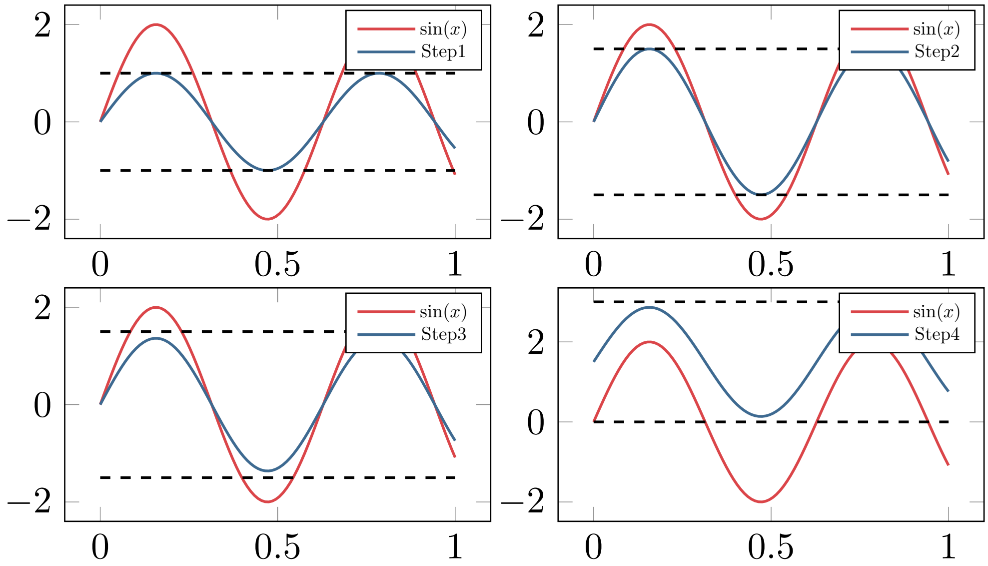


Figure 3.13: ADC signal conversion procedure: In Step 1 the input signal is converted in per unit, in Step 2 the signal is mapped in the range ±1.5, in Step 3 the signal is multiplied by a gain 1/1.1 to allow that the input signal exceeds its nominal value, in Step 4 an offset is added to make the signal positive

,

The conversion is shown in (3.10).

$$V_{\text{out}} = V_{\text{in}} \frac{\alpha_{\text{ADC}}}{\alpha_{\text{PU}}} \frac{1}{\gamma} + \Phi \tag{3.10}$$

where:

- V_{in} is the measure to send to the controller
- V_{out} is the output voltage
- α_{ADC} is fixed and is $V_{\text{ref}}^{\text{hi}} - V_{\text{ref}}^{\text{lo}} = 3V$ where $V_{\text{ref}}^{\text{hi}}$ and $V_{\text{ref}}^{\text{lo}}$ are two parameters that it is possible to find in the data-sheet [24].
- α_{PU} is fixed and is 2
- γ is the tolerance over the limit in our case 110% = 1.1 of the nominal value in per Unit.
- Φ is the offset used to make the signal positive and it is equals to 1.65V, i.e. the half of the ADC voltage range.

Virtual Analog Input: referring to Figure 3.11 let's analyze the last signal, i.e. the I_{dc} current. In this case an Analog In to the simulator is required, but as explained before, the interface board doesn't have this connector. The only suitable solution is to modulate the I_{dc} signal and produce a PWM. A *PWM capture block* in PLECS capture the PWM signal and then it is filterd with a digital low pass filter. This solution introduces a small delay at about $0.0018s$ due to the filter, but it is negligible for our purpose. We use a first order filter with the transfer function shown in (3.11).

$$H(s) = \frac{1}{1 + sT_c} \tag{3.11}$$

Here $Tc = 0.0008$ is the time constant of the filter. The cutoff frequency calculated as shown in (3.12).

$$f_{\text{cut-off}} = \frac{1}{2\pi T_c} = 198.94\text{Hz} \tag{3.12}$$

After converting the filter in discrete time with MATLAB using as sample time the same of the RT Box ($Ts = 5\mu s$) the final transfer function is shown in (3.13).

$$Hd(s) = \frac{0.0018}{z - 0.9876} \tag{3.13}$$

3.2.4.1 Synchronization

An important aspect to consider when different equipment is used, is the synchronization. The real-time simulator and the control board have two different clocks, which are not synchronized together. When they exchange signals this might cause noise or errors as explained for the Digital Input block in the previous section. Fortunately, it is possible to synchronize the control baord with the RT Box. The idea is to generate a square wave with the real-time simulator and send it to the control card. A spacial pin of the control card captures the signals and synchronizes adequately the EPWM modules and the ADC acquisition. Figure 3.14 shows a conceptual diagram of the synchronization process.

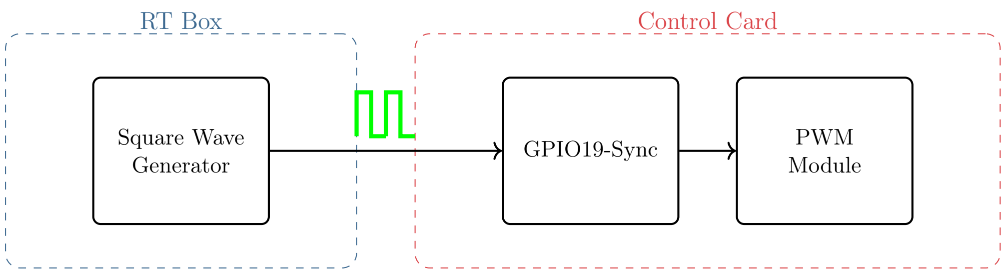


Figure 3.14: Synchronization mechanism

Using this strategy the PWM Capture blocks are not required any more and we can use simply the Digital Input blocks. For computation intensive applications this could be crucial because we noticed that using the Digital Input blocks instead of the PWM Capture blocks reduces the CPU load of the simulator with approximately 5%.

3.2.5 Texas Instruments LaunchPad LAUNCHXL-F28379D

The LaunchPad LAUNCHXL is a development board that incorporates the chip F28379D. The micro-controller is a high performance dual core unit with a 200MHz clock.

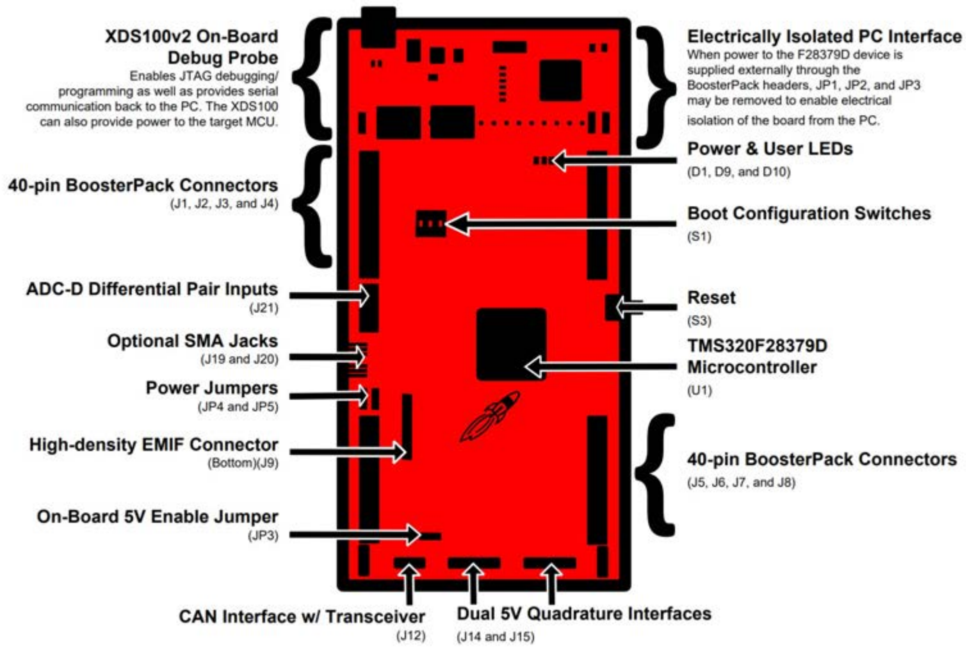


Figure 3.15: The LaunchPad main features [24].

Figure 3.15 shows the features of the module. It offers JTAG debug interface, serial communication, SPI, SCI, I2C, UART, and also a series o GPIO pins that allows to receive and transmit digital signals. Some of them are PWM and interrupt enabled. The board has 16 ADC channels that are perfect for our application as several of analog inputs and PWM digital outputs are required. The board presents an electrically isolated interface. When the board is supplied externally (in our case by the real-time simulator) and not through the USB port it is possible to remove specific jumpers to enable electrical isolation from the board to the PC.

A pin-out representation of the control board is shown in the Figure 3.16.

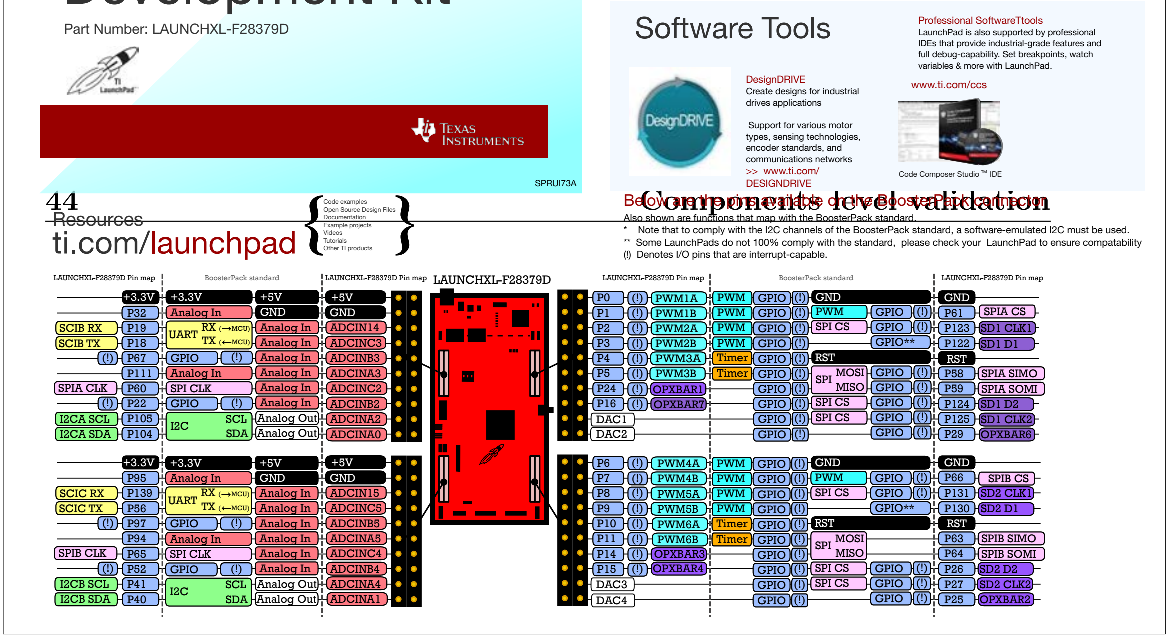


Figure 3.16: The diagram shows the LaunchPad IOs pins and their functions [24]

Referring to the manual, the micro-controller has some useful modules. The ePWM and ADC are the most important and they are essential for our project.

3.2.5.1 ePWM module

The pulse width modulation technique (PWM) is used in many applications because produces a digital approximation of an analog signal. According to the definition, the PWM signal consists in a sequence of pulses with variable width and constant amplitude. An important property of the PWM is that the pulses contain the same energy of the original analog signal so the technique is perfect for controlling the converter of our project [24]. The F28379D chip has 12 modules called ePWM; each module produces two independent outputs (EPWMxA, EPWMxB) that can be used for multiple purposes. Different modules can be synchronized together using a master and slave configuration but also by receiving an external signal. Additionally the module outputs can be used to generate interrupts to start the ADC conversion.

In our project this module plays a fundamental role. All the PWM signals used to control the MOSFET gates are synchronized together with a frequency of 10KHz. At the same time the module generates the ADC start conversion signals and triggers the interrupt routine in which the controller is updated. The strategy of the periodic interrupt routines is crucial for a real time system because it allows the controller to be updated with a fixed frequency.

3.2.5.2 ADC converter

In many application the sensors produce analog signals that must be converted to a digital value to be stored and elaborated by a microprocessor. In the circuit of a power electronics converter there are several sensors that monitor the voltages and the currents of the device. All these signals are analog signals so multiple analog to digital converters are required. The F28379D has four ADC modules, with an input range of 0-3V. A dc/ac converter usually works with low/medium voltages (1KV-10KV) so to make the measurements compatible with the range of the microchip, dedicated circuits are used. These circuits reduce the voltages with transformers ensuring galvanic isolation. In our case we use the RT Box that thanks to the configuration explained before does not require additional power amplifiers or apposite interfaces.

Each ADC module has six channels and it can works in two different ways, in the differential mode where a pair of pins (A+, A-) is sampled simultaneously with a resolution of 16 bits and the result is the difference between the two sampled values. The maximum conversion performance is 1.1 MSPS (Mega samples per second) and the resolution is around $15\mu V$. In the single-ended mode the resolution is 12bit with a conversion performance of 3.5 MSPS. The advantage of the differential mode is the ability to cancel noise but it requires the double of the channels in comparison to the single ended mode.

To setup the ADC modules there are some important aspects to consider, i.e. how to interpret the result, how to set the acquisition windows and how much time the microcontroller requires to convert the signal.

When the ADC finishes the conversion it stores the value in a register. The single ended mode is used in this project, ensuring a resolution of 12 bit that can be interpreted as a range of $0 - 2^{12} = 0 - 4095$ possible values. If the analog signal is below the $V_{\text{ref}}^{\text{lo}}$ (that is the zero reference) the results is zero, if the input is above $V_{\text{ref}}^{\text{hi}}$ (that is the reference for the upper limit of 3V) the result is 4095. If the input is

$$V_{\text{ref}}^{\text{lo}} < V_{\text{ADC}} < V_{\text{ref}}^{\text{hi}} \tag{3.14}$$

the output has a value according to equation 3.15.

$$O_{\text{out}} = 4096 \frac{V_{\text{ADC}} - V_{\text{ref}}^{\text{lo}}}{V_{\text{ref}}^{\text{hi}} - V_{\text{ref}}^{\text{lo}}} \tag{3.15}$$

Therefore, to obtain again the result in volts we have to arrange the equation as shown in 3.16

$$V_{\text{in}} = V_{\text{ADC}} = V_{\text{ref}}^{\text{lo}} + (V_{\text{ref}}^{\text{hi}} - V_{\text{ref}}^{\text{lo}}) \frac{O_{\text{out}}}{4096} \tag{3.16}$$

Here, V_{in} is the input voltage to the ADC converter coming from the RT Box so to obtain again the value in per unit we invert (3.10) obtaining 3.17

$$V_{\text{in}} = (V_{\text{out}} - \Phi) \frac{\alpha_{\text{PU}}}{\alpha_{\text{ADC}}} \gamma \tag{3.17}$$

The equation closes the loop of the signal transmission between the micro-controller and the simulator. In the next step the acquisition time and the acquisition windows will be presented. For a 12 bit resolution conversion the acquisition time is fixed and corresponds to 10 clock cycles. The resolution influences also the acquisition windows. Each module has a Sample and hold circuit shown in Figure 3.17.

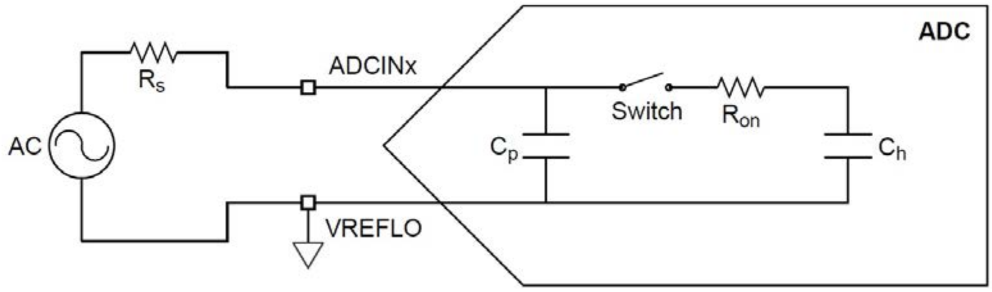


Figure 3.17: Single-ended input diagram [24]

The C_h capacitor requires time to be charged so it is important to know the minimum acquisition window of the ADC. The user has to choose the sample and hold duration to ensure that the capacitor has enough time to be charge, in order to guarantee a proper resolution in the conversion. In our case we have a resolution of 12 bit and knowing that the ADC range is 0-3V we can express the resolution in volts as shown in 3.18.

$$V_{\text{res}} = \frac{\alpha_{\text{ADC}}}{2^N} = 7.32 \times 10^{-4} V \tag{3.18}$$

where $N = 12bit$ and $\alpha_{\text{ADC}} = 3V$. The manual [24] suggests to choose the sample and hold time so that the capacitor voltage has enough time to charge and stabilize around the ADC input voltage with an accuracy LBS (Last Significant Bit) of 0.5 or 0.25 of V_{res}. An estimation of the sample and hold time can be obtain using 3.19.

$$t = -\ln \frac{\text{LBS}}{2^N} T \tag{3.19}$$

where T is the time constant of a RC filter with $R = R_s + R_{ON} = 2K$, $C_h = 4.5\text{pF}$ and LBS $= 0.25$ is the accuracy. The result is $t = 87ns$, which corresponds to about 17 CPU clock cycles. This value can be computed as shown in 3.20.

$$N_{\text{ACQ cicles}} = \frac{t}{f_{\text{CPU}}} = \frac{87ns}{200Mhz} \approx 17\text{CPU cicles} \tag{3.20}$$

The minimum windows acquisition time is computed as shown in 3.21.

$$T_{\text{ACQ}} = \frac{(N_{\text{ACQ cicles}} + 1)}{f_{\text{CPU}}} = (17+1)/200MHz = 90ns \tag{3.21}$$

T_{ACQ} is very small compared with the timing of our application so the conversion and acquisition of the signal is fast enough. Knowing the limits of the system is very important because, it allows us to understand if the hardware is powerful enough to satisfy all the project requirements.

Before using the ADC modules it is important to perform a periodic calibration. According to the manual there is a dedicated register (ADCOFFTRIM) where the offset is stored. The procedure consist in three main steps:

- Add an artificial offset (e.g ADCOFFTRIM = 100);
- Perform multiple acquisition of the GND reference of the RT Box and compute the average V_{AVG};
- Set the ADCOFFTRIM = $100 - V_{\mathrm{AVG}}$.

in this way the zero offset is removed.

For the project I trigger the ADC modules with the same PWM (EPWM4A) that controls the start-of-conversion blocks (SOC). In the SOC the user choose three parameters, the channel to convert, the trigger source and the acquisition window. With this configuration it is possible to obtain a snapshot of all the measurements at the same time, avoiding computational errors due to an asynchronous acquisition of the measurements. Once the system is correctly configured the last thing to consider is the data acquisition.

3.2.5.3 Serial communication

The LaunchPad has a JTAG debugging interface that allows to monitor all the variables of the system and the memory addresses. The debugging interface allows to modify in real time the variables and the addresses, for example, for tuning the controller or enabling/disabling flags in the code. However this method cannot be used to collect data. The refresh rate is very low and for our application we are interested to see the dynamic of the controller that usually are very fast. In debugging mode the refresh rate is not constant and it has a frequency of about 1Hz. The transients in our system instead last 100ms or less, so a faster sample rate is required but unfortunately it cannot be achieve with any communication interface. For example the RT Box allows to collect data via UDP with a maximum frequency of 1000 Hz and also this fast communication type is not enough. The only possible solution is to store the data locally and then send it via the communication protocol which the user prefers. The control card provides serial communication with all the necessary libraries.

The Serial Communication Interface (SCI) is commonly used to exchange information between devices, it is asynchronous and not so fast but can communicate over long distances. In our case it is important to have flexibility because not always the control card is close to the PC so having a long (2-3m) cable it's a thing to consider.

The SCI uses only two pins, TX to transmit data and RX to receive data. The maximum band rate supported by the F28379D chip is 115200 bits/sec which for our purpose is enough. Fortunately, the control card has the SCI pins mapped to a FTDI chip that is a serial USB converter so it is possible to directly connect a mini USB cable to the LaunchPad to receive the data.

3.2.5.4 Data acquisition

To record the dynamic of the controller a suitable sample rate is in the order of 2-10 KHz and the only way to achieve this sample rate is to store the data in RAM and then send it via SCI. How much data is it possible to store in the RAM?

The RAM memory of the F28379D chip is divided in blocks as shown in the Figure 3.18:

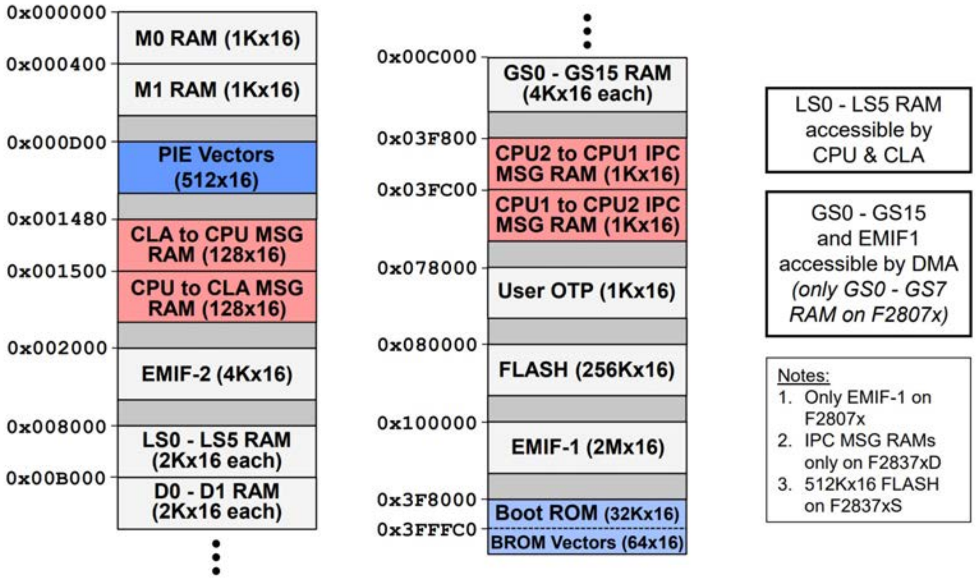


Figure 3.18: Memory map

The blocks M0, M1, LS0-LS5 are already used by the system to store the global and constant variables, to allocate stack space, for *"malloc"* type functions and so on. The only free blocks are the GS0-GS15. This part of the memory is shared between the two CPU cores so both of them have access to the data stored in it. Each GSx block has the size of 4Kx16 words and knowing that a word is composed of 16 bit, the total free memory is:

$$m_b = 4K * 16 * n_b = 4096 * 16 * 16 = 1\text{Mbit} \tag{3.22}$$

where n_b is the number of GSx blocks.

Our goal is to store a window of 5-10sec of at least 4 variable with a sample rate of 2KHz to monitor the frequency, the active power, the voltage and current magnitude. Saving the data in a float format, requires 32bit for each value, and with a sample rate of 2KHz it is possible to store a window of 4.096s that is not enough for our purpose as shown in (3.23).

$$t = \frac{1Mbit}{32bit * 2Khz} \frac{1}{n_v} = 4.096sec \tag{3.23}$$

where n_v is the number of variables to store.

A possible solution is storing the data in a fixed-point format. The number of bits used for the fractional part determines the resolution. The minimum acceptable resolution is of 3 digit after the comma. That means we need 9 bits to store the fractional part with the required resolution as shown in the following equation

$$R = \frac{1}{2^9} = \frac{1}{512} = 0.001953 \tag{3.24}$$

Considering the integer part, the measurements stay inside the interval 0-70, because the controller works in per unit and the only value bigger than 1 is the frequency that usually stays in this interval, which can be represented by 7 bits ($2^7 = 128$). So the total number of bits considering the fractional and integer part required to represent the data are $9 + 7 = 16$ bits, the same space that occupies a word. The acquisition window with the fixed-point format is shown 3.25.

$$t = \frac{1Mbit}{16bit * 2Khz} \frac{1}{NumberOfVariables} = 8.192sec \tag{3.25}$$

that is enough for out purpose but it presents technical difficulties because the programming language doesn't natively support the fixed point format and for that reason it is necessary to define new data types and implement all the functions required to convert and treat this type of data. However, following the same procedure we can obtain a similar result that is easier to implement.

An alternative consist of multiplying each value by 1000 and then storing the value in a short variable that occupies 16 bits and has a range of $[-32767, +32767]$. The last problem to solve is finding a way to represent the frequency because for example if $f = 50.123Hz$ applying the multiplication produces a results out of the range. A possible solution consist of subtracting the reference value, i.e. 50Hz or 60Hz, and sending only the difference as shown in 3.26

$$\Delta f = f - f_{\text{ref}} = (50.123 - 50.000)Hz = 0.123Hz \tag{3.26}$$

Then the receiver can easily recover the initial value by adding the reference value of the frequency. This method allows to store and send data efficiently with the same performance of the fixed-point method, reducing a lot the complexity of the code. Base on the specification of the system it is possible to compute a trade-off between the acquisition window, the number of signals to send and the sample rate to match the specifications of every particular application. For example if we want to send more signals we have to reduce the acquisition windows or to decrease the sample frequency. For our project a sample rate of 2 KHz it enough because guarantees an acquisition windows of around 8 seconds, allowing to send all the necessary measurements.

The data has to be stored in a particular section of the RAM (GS0-GS15) so it is necessary to modify the memory map of the F28379D chip. In the main file we add the following code:

```
#define RESULTS_BUFFER_SIZE 65536
short data[RESULTS_BUFFER_SIZE];
#pragma DATA_SECTION(data, ".DATA_TABLE");
```

Listing 3.1: main.c

that specifies the allocation of the data array in the RAM memory. In the *2837xD_Generic_FLASH_lnk.cmd* we commented out all the RAMGSx blocks and we added the code shown in Listing 3.2.

```
SECTIONS
{
...
.DATA_TABLE: > RAMGS, PAGE = 1
}

//Note that here I redefined 'RAMGS' to be one chunk that spans all 16 GS blocks:
PAGE 1 :
RAMGS : origin = 0x00C000, length = 0x010000
//RAMGS0 : origin = 0x00C000, length = 0x001000
//RAMGS1 : origin = 0x00D000, length = 0x001000
...
```

Listing 3.2: Memory allocation.

In this way the data array is allocated correctly in the block RAMGS that can be after transmitted via SCI. An interesting value is the transmission time that is around 4.8 seconds, the data transmission has a lower priority than the ADC interrupt routine so the data transmission doesn't affect the controllers. In the next section we will see how to receive the data and interpret it.

3.2.5.5 Node-RED

Node-RED is a programming tool based on a browser-based editor used to easily link hardware devices, APIs and online services [28]. It works on multiple platforms (Windows, Linux, MacOS), and due to its light routine based on Node.js can runs on limited hardware like a RaspberryPi. Node-RED provides a wide range of nodes that can be connected together, producing a sort of flow chart. Each node implements a function or establishes for example a communication protocol giving to the user a powerful tool to connect with hardware using a high level abstraction.

For this project the platform is used to collect the data coming from the LaunchPad developing board and store it in a file. Considering the Figure 3.19, a serial communication node establishes the communication with the F28379D chip, the incoming data is parsed and stored in a circular array. When the array is full it sends the received data to a *CSV* node that converts the data in the CSV format and saves it in a CSV file. When the process is completed, the buffer array is cleaned and ready to receive new data.

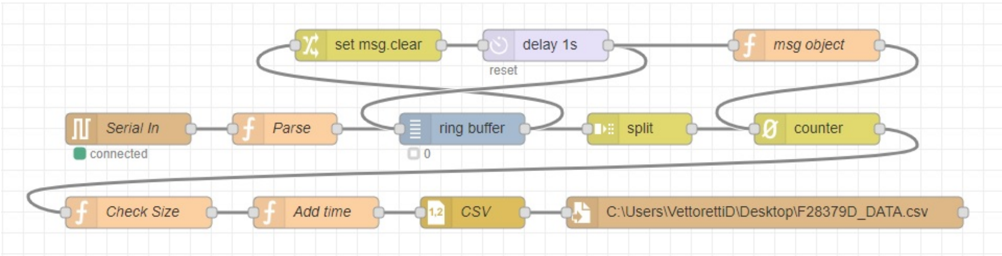


Figure 3.19: Node-RED flow

Node-RED has also the advantage of having a dashboard which allows to display data in real time. For example, if we are interested in monitoring data, we can send a new package every second with the LaunchPad and then represent it in a real time graph. An example is shown in Figure 3.20, where the frequency, the power, and the voltage magnitude are shown with a refresh frequency of 1Hz. We configure the dashboard for an easy interaction with the control card and to constantly monitor the state of the system. As future work it is possible to implement several functions, e.g the user can use a touchscreen to set the among of active and reactive power or connect and disconnect a load to the grid enabling a toggle button.

3.2.5.6 Scripts to automatically perform a CHIL simulation and data acquisition

The project is composed of multiple platforms that work separately, so when the user runs a simulation he has to configure each system and set multiple parameters which makes the procedure cumbersome. To simplify the process I wrote a script in Python that automatically starts the simulation and acquires the data. First of all let's introduce the main steps necessary for running a CHIL simulation:

- Turn on the LaunchPad, which automatically starts the controller.
- Open Node-RED, which directly starts the serial communication with the LaunchPad.
- Load the model in the RT Box and run it.

Figure 3.20: Node-RED Dashboard

- Simulate a load step or any other required operation.
- Start the data acquisition;
- Acquire the data and save it.

From the point of view of the LaunchPad the program resides in the flash memory so as soon as the board is turned on, the controller starts to run. In the same way Node-RED automatically establishes the serial communication with the LaunchPad. The crucial missing part is a system that links the other parts synchronizing everything together. The RT Box fulfills this task providing a very flexible solution that completely automatizes the CHIL simulation.

For example consider that our goal is to simulate a load step in real-time. To do it, an additional load must be connected to the grid, as shown in Figure 3.21. The simple way to connect it, is to externally control a three phase switch that connects or disconnects the load to the grid. The external command can be sent from a Python script. An special PLECS block, the "Programmable Value" block receives the external signal and closes or opens the switch. In the same way, the external signal can be use to trigger the data acquisition inside the LaunchPad. The "Programmable Value" block is routed to a "Digital Out" block that sends the command (a Boolean 0-1 value) to the LaunchPad. The raising/falling edge of the signal is managed by an interrupt from the Launchpad that triggers the acquisition. Usually, the load step is centered in the acquisition windows, this behaviour can be achieved by adding a delay in the switch signal, as shown in Figure 3.21. This procedure allows to synchronize the data acquisition and the LaunchPad with the RT Box.

The last part to consider is the Python script that is presented in Listing 3.3.

```
1  import xmlrpclib
2  import time
3
4  #Set ip initialize server and
5  ip = "192.168.0.100"
6  server = xmlrpclib.Server("http://" + ip + ":9998/RPC2")
7  path = "GF_droop_Z_REAL.elf"
8
9  #Load the inverter model
10 with open(path, "rb") as f:
11     print("Uploading executable")
12     server.rtbox.load(xmlrpclib.Binary(f.read()))
13 f.closed
14
15 #Start execution
16 print("Starting executable")
17 server.rtbox.start()
18 print("Simulation running")
19 time.sleep(10)
20
21 print("Collecting data..")
22
23 #Enable data acquisition
24 server.rtbox.setProgrammableValue('EnableAcq', [1])
25
26 time.sleep(15)
27 server.rtbox.setProgrammableValue('EnableAcq', [0])
28
29 #Stop execution
30 print("Stopping executable")
31 server.rtbox.stop()
```

Listing 3.3: Python example

The producer of the real-time simulator provides a library used to establish a TCP-IP communication with the RT Box. After the communication is opened the script loads the circuit model (an *.elf* file produced during the model compilation) and starts the real-time simulator. At this point, the data acquisition command is sent with the instruction at row 24. After 15 seconds the data acquisition ends and the script sends a command to stop the CHIL simulation.

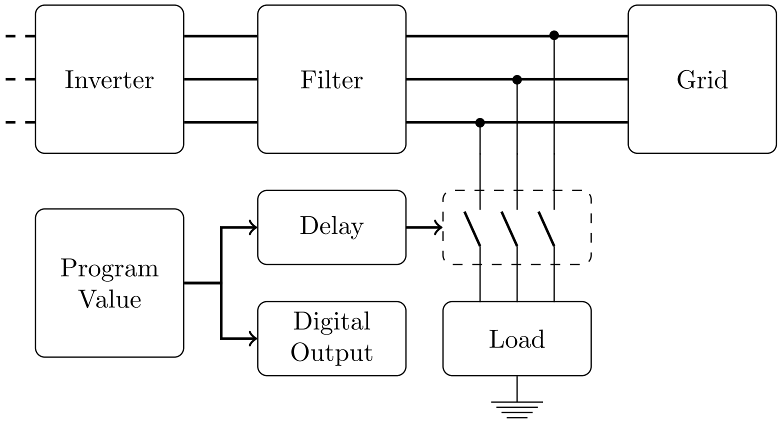


Figure 3.21: The *Program Value* block receives a command from a Python script. This command is sent trough a *Digital Output* block to the control card to trigger the data acquisition. In the meanwhile, the same signal closes the breaker connecting a load to the grid. The delay is used to center the load step in the acquisition window.

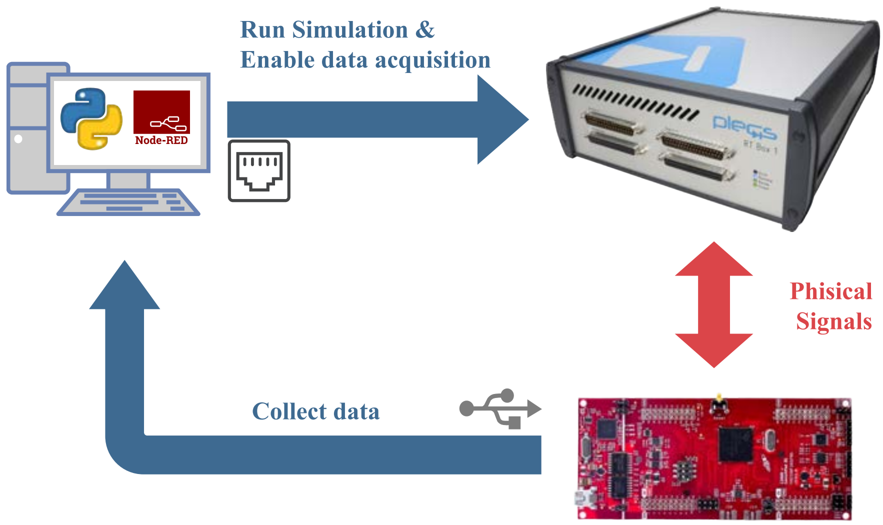


Figure 3.22: Data acquisition schematic representation

3.3 Use Cases

Lately, the interest of both academia and industry has started to focus on in the validation of the grid forming strategies presented in Chapter 2. The setup presented in this project plays a fundamental role in the study of how these new techniques behave when they are implemented in real hardware.

For that reason we will perform a CHIL simulation trying to verify if the results obtained in the offline simulations are comparable with the Real-Time simulations. For each technique an experiment will be performed, where the hardware under test is the one presented in Figure 3.10, and the controllers implement the Droop, the dVOC, the Matching, and the VSM strategy. During the simulation a constant impedance load will be connected to the network, increasing the converter energy demand with 0.25 pu. The Python script presented before will be used to automatically run the simulation.

As for the offline simulations, it is important to verify if the system is stable. Since the grid forming strategies have to form the grid and not just inject or absorb power, the main parameter considered to evaluate the converter stability is the frequency of the network. Given a frequency reference of 50Hz, the controller has to respond adequately to load changes, avoiding large overshoots. Furthermore, it is required that the controller response must be fast enough to avoid that the grid frequency collapses. In the scientific community several offline simulations have already been performed, showing good results in term of controller stability and behaviour in response to load chances. With the CHIL simulations we are interested to observe the behaviour of the controllers when there are hardware limitations and delays in the signals transmission. After simulating the system and collecting data, the obtained results for each control strategy are shown in Figure 3.23, 3.24, 3.25, 3.26.

In addition to the frequency, also the power of the converter, the voltage magnitude of the network, and the current magnitude flowing inside the load are taken in account. For all four strategies the behaviour is comparable. Starting from the frequency plots it is evident how the undershoot is larger in the real case compared to the offline simulation. This means that the CIL has a slower response due to the delays introduced by the signals transmission. These delays are produced by the ADCs of the RT Box and the LaunchPad board. In particular for the Matching strategy this behaviour is accentuated because the frequency directly depends on the V_{dc} voltage so the delay introduced by the Low-Pass filter that produces the current reference of the current source has a major contribution to the frequency dynamics. Another aspect to consider is that more noise is present in the CHIL simulations with respect to the offline simulations, particularly for the voltage magnitude.

Beside these two aspects, the CHIL simulations reflect what we have observed also in the offline simulations, i.e. Grid Forming strategies can properly form a grid and remain stable under the simple use cases that were tested.

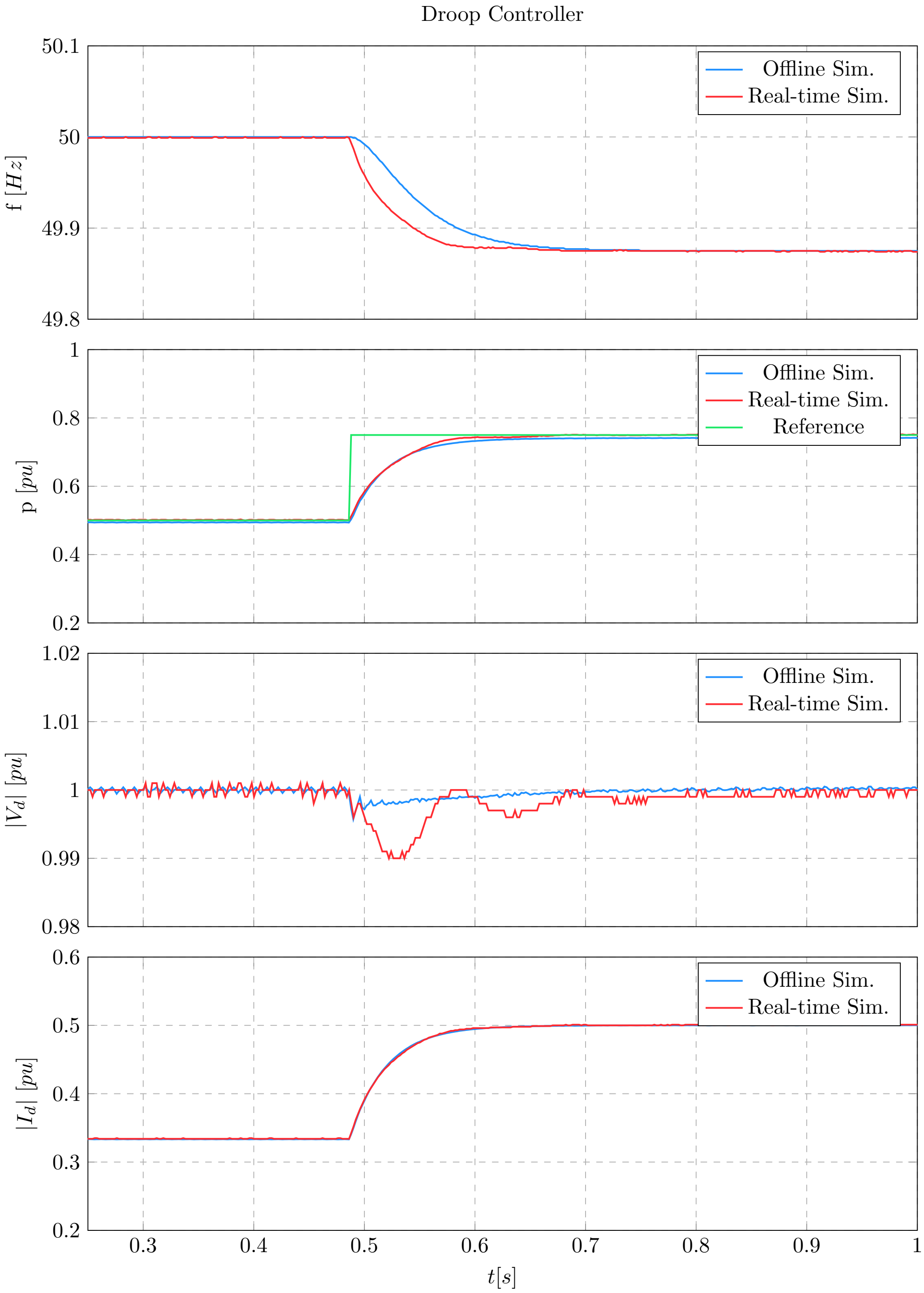


Figure 3.23: Droop strategy: Comparison between offline and real-time simulation. The disturbance is the connection to the network of a constant load of 0.25 pu of the nominal power of the converter.

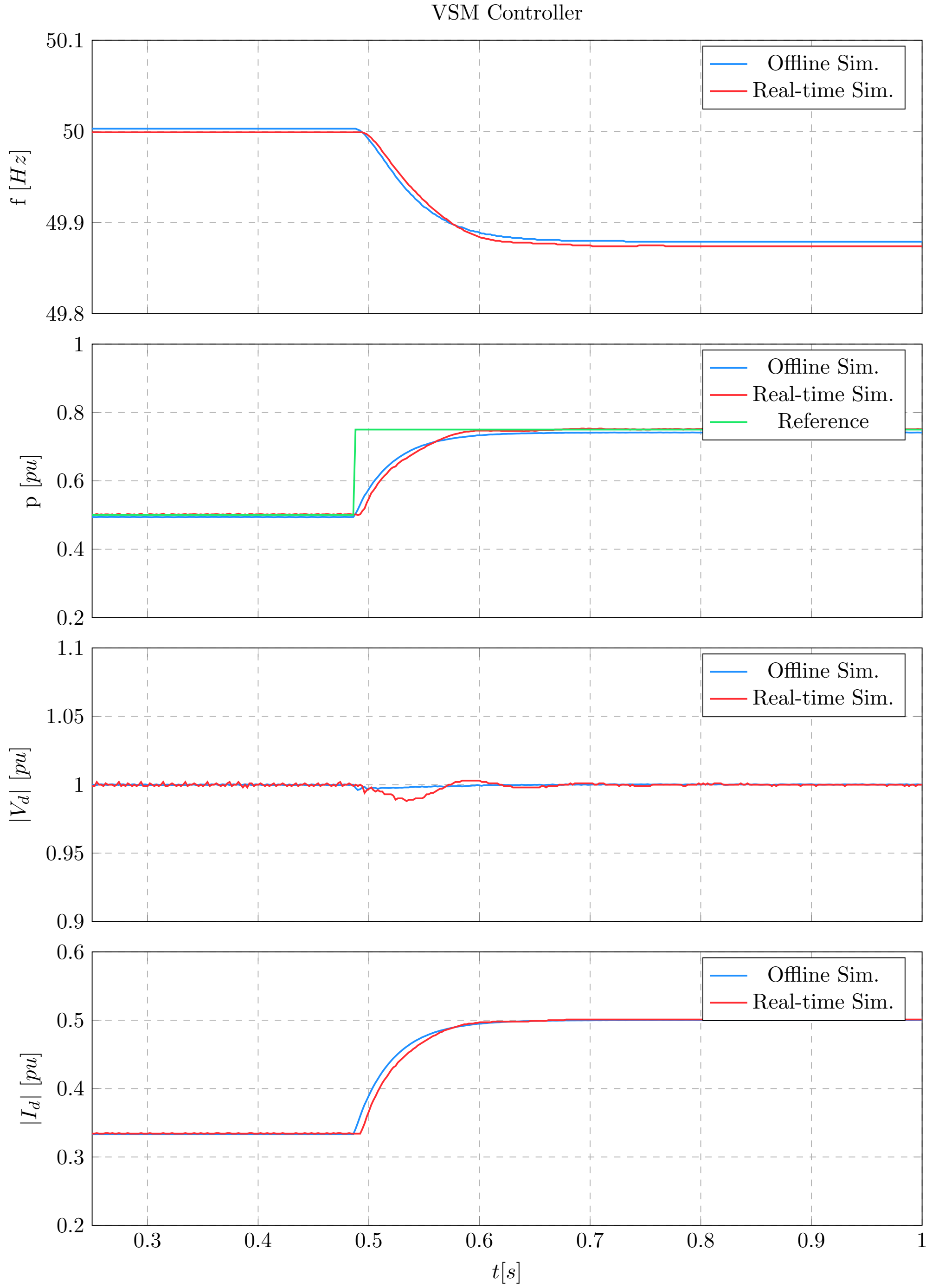


Figure 3.24: VSM strategy: Comparison between offline and real-time simulation. The disturbance is the connection to the network of a constant load of 0.25 pu of the nominal power of the converter.

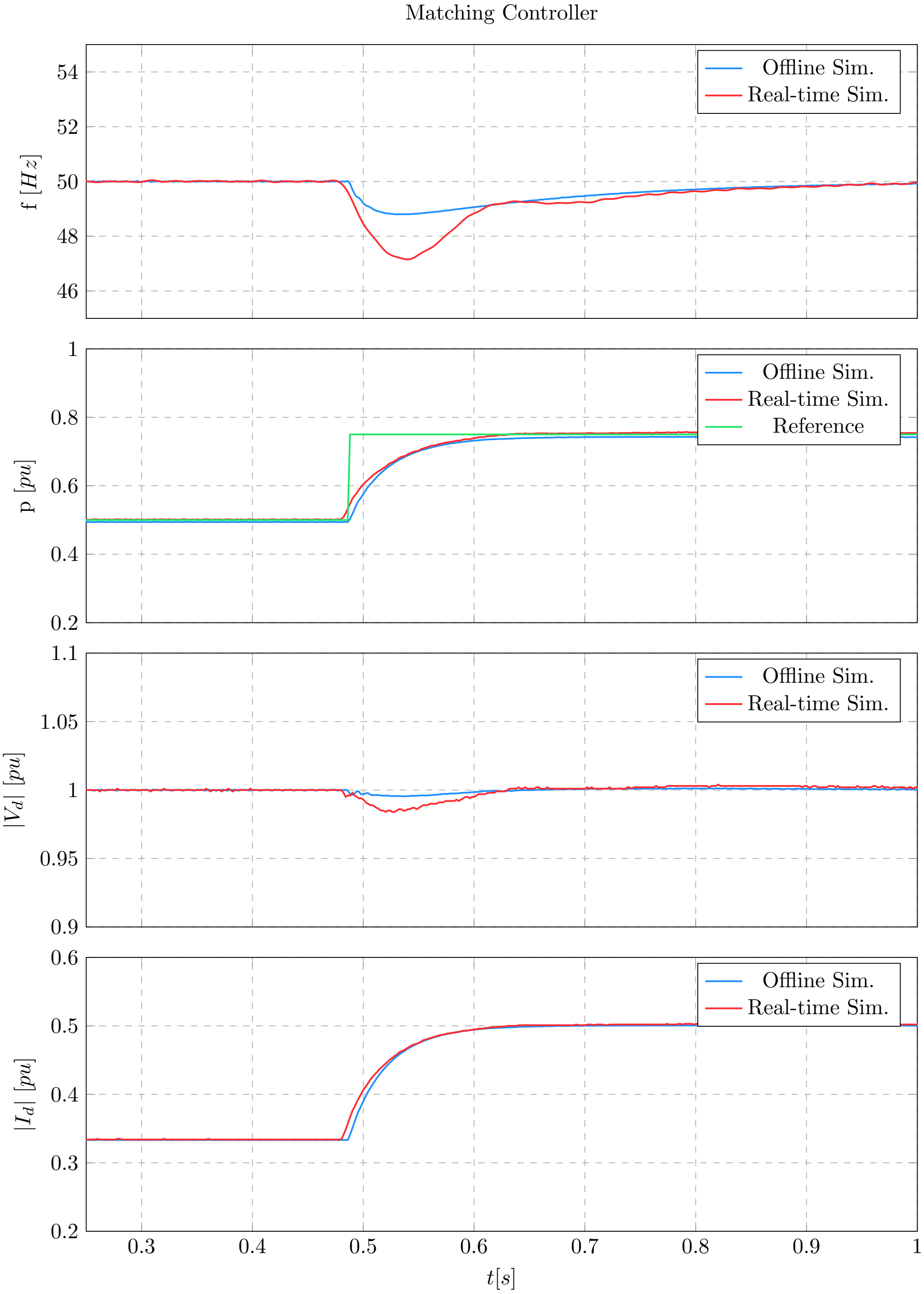


Figure 3.25: Matching strategy: Comparison between offline and real-time simulation. The disturbance is the connection to the network of a constant load of 0.25 pu of the nominal power of the converter.

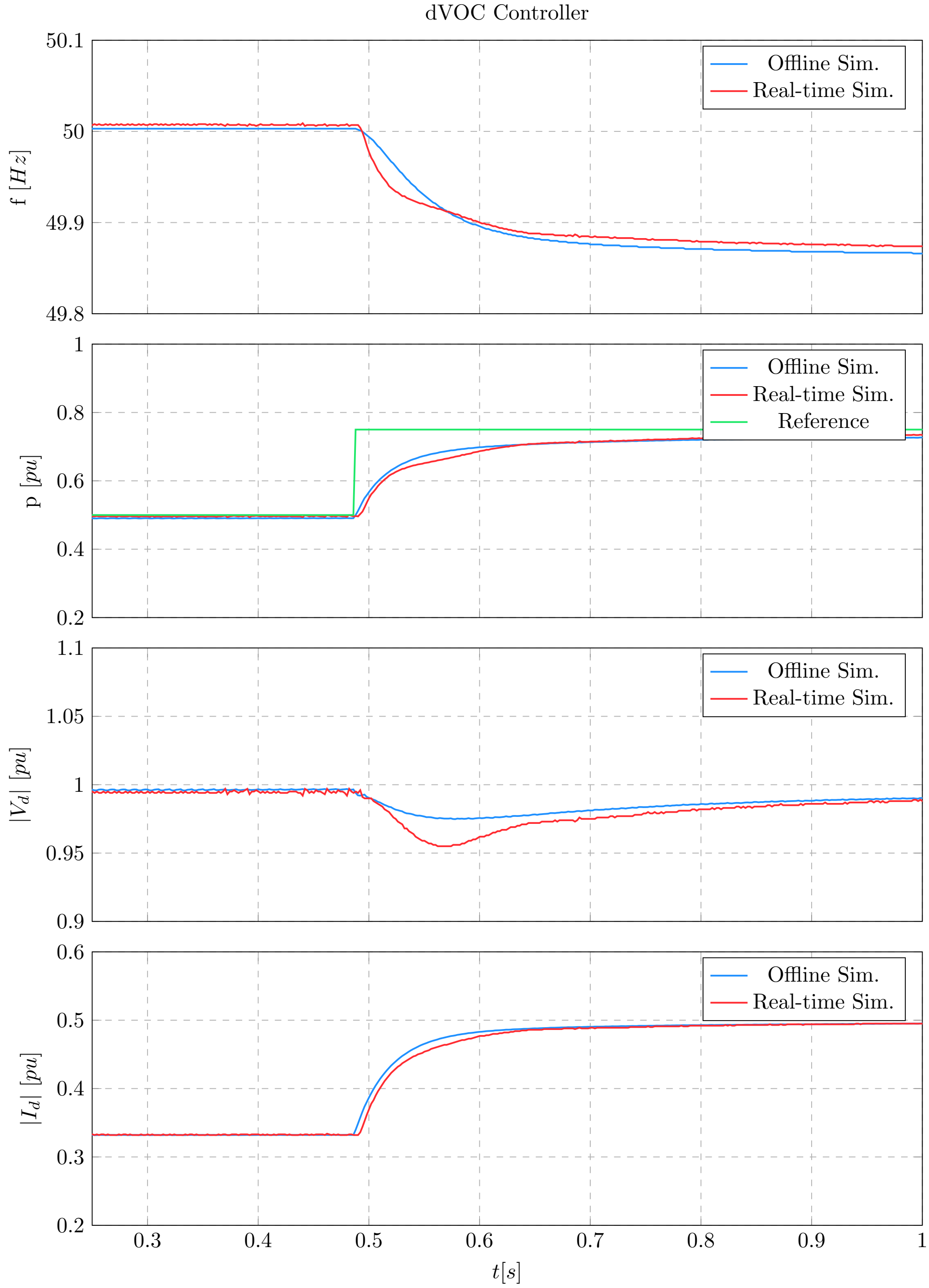


Figure 3.26: dVOC strategy: Comparison between offline and real-time simulation. The disturbance is the connection to the network of a constant load of 0.25 pu of the nominal power of the converter.

3.4 Conclusions

While in the previous chapter we introduced the grid forming control strategies from a theoretical point of view, this chapter was focused on the component level validation of the grid forming controllers. Our goal was to build a setup able to perform CHIL simulations in order to test the control algorithms and study their behaviour when they run on real hardware. After introducing the idea of CHIL simulation, which requires a real-time simulator and a control board connected together through an interface card, each component of the setup was analyzed. Stating form an overview of the real-time simulator describing its architecture and features, we proceeded to defining the methodology used to build the models and the controllers under test. The chapter follows with a detailed explanation concerning the control card and all the challenges addressed to solve synchronization problems with the real-time simulator. In addition, we propose different solutions to optimize the code in order to increase the performances of the control card. The simulation execution and the data collection were made automatic procedures to simplify the final process for carrying out the CHIL simulations. Finally, we successfully implemented all the four grid forming control strategies in the control card. After running the CHIL simulations we collected the data and compared the results with the offline simulations, showing comparable results in terms of controller response.

Chapter 4

System Level Validation

This chapter is focused on System level validation. Our purpose is to setup a platform which is able to validate the grid forming techniques presented in Chapter 2 in a wider scenario, but with the same accuracy level used for the component level validation approach. The chapter starts with a description of the setup used to perform system level validation. We consider a new and more high performance real-time simulator (OPAL-RT), subsequently, follows a detailed description of how to coordinate and configure the different hardware parts to make sure that they work together. The chapter ends with the description of a custom interface board developed to connect the real-time simulator with a custom control card

4.1 Introduction

The grid forming techniques presented in the Chapter 2 have been successfully implemented in the control card and tested using CHIL simulations. Until now we performed component level validation, where the hardware under test was the control card. Now we move our attention to a scenario where multiple control cards are connected to the real-time simulator in order to test a larger network. Running real-time simulations on a such network goes under the name of system level validation. What is the purpose of using a system level validation approach? With component level validation, we tested the behaviour of the grid forming controllers that are implemented into the control card. In this scenario the converter was connected to a simple network and we focused our attention on the problems introduced by the delays and the noise in the signal transmission and the ADC resolution. With the system level validation approach instead, multiple grid forming converters and synchronous machines are connected together forming a complex structure that represent better a real network. Therefore, with this kind of setup it is possible to test the penetration of renewable energies, showing how low inertia networks composed by wind turbines and PV panels influence the stability of the grid. Multiple load steps can be tested but also the disconnection of various synchronous converter or generators. All of that can be tested with CHIL simulations keeping the same level of accuracy adopted for the component level validations. In relation to what is just introduced, the development of a setup which is able to simulate a wider grid plays a fundamental role in the studying and validation of the new grid forming techniques, and it is also the main topic of this chapter. A setup used to perform system level validation is a complex system composed of multiples parts connected together. A general representation of it is shown in Figure 4.1. The setup is built following a modular approach. Each basic module is composed by a control board interfaced with the real-time simulator, presenting the same structure used to perform component level validation introduced in Chapter 3. The real-time simulator in this case is a more high performance unit able to emulate multiple inverters and generators. In the next sections we explain in detail the architecture of the simulator and the development of an interface board used to connect a custom control card to it.

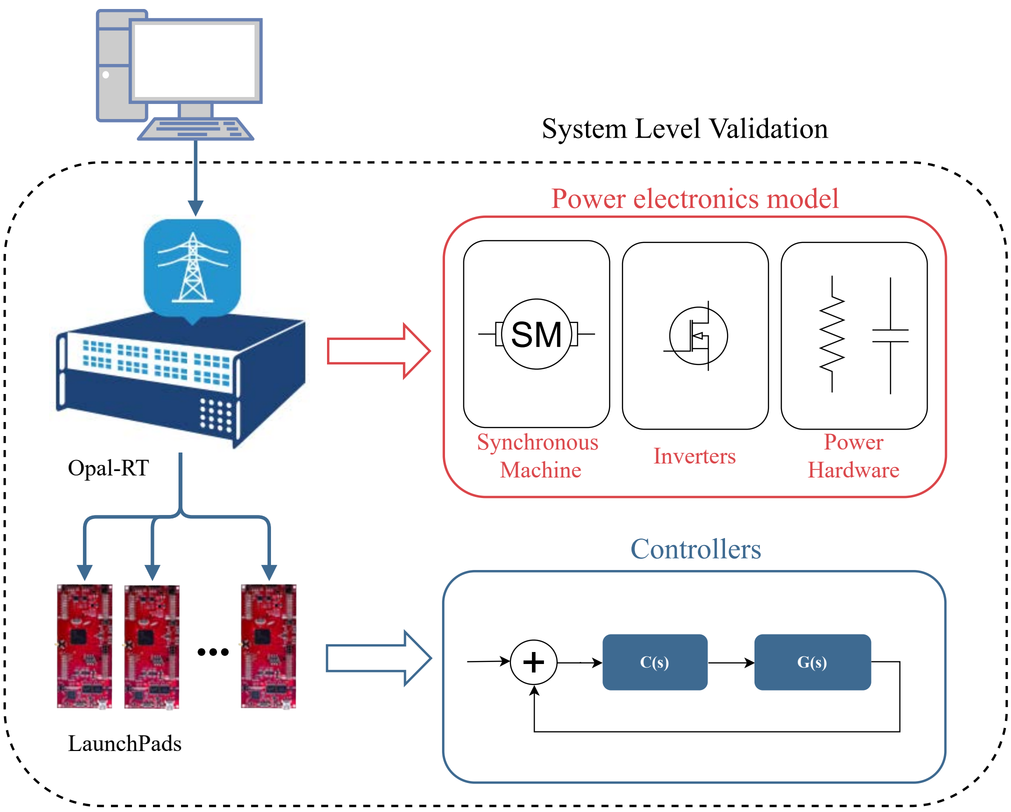


Figure 4.1: Conceptual representation of the System level Validation setup

4.2 OPAL-RT

Due to the high computational power required for simulating a scenario composed of multiple power electronics converters connected to a complex electrical network, the RT Box presented before cannot be used for our purpose, because it is not powerful enough. For that reason, it is necessary to consider another real-time simulator that is specifically developed for this scope. OPAL-RT is a powerful FPGA-Based real-time simulator which can be used for CHIL simulations, but also for fast control prototyping. It combines the power of a 8 cores Intel Xeon processor with a Xilinx Virtex-7 FPGA being able to satisfy the requirements of the most demanding applications [29]. The three main advantages compared to the RT Box are:

- the power and the performance: it benefits of parallel processing units to simulate large and complex systems and an FPGA to simulate power electronic devices;
- the connectivity: OPAL-RT has 256 high-speed digital and analog I/O lines and 16 fiber optic SFP sockets.
- the expandability: it is possible to connect multiple OPAL-RT simulators using special expansion units through PCI-Express.

To understand the key features of OPAL-RT it is necessary to introduce its architecture. In order to have a clear idea of how it works, this discussion will be presented in the following section.

4.2.1 OPAL-RT architecture

The architecture of the real-time simulator is presented in Figure 4.2. On the left side one of the main components, the Real-Time PC, is shown . It is based on a Linux operating system that runs on a computer that integrates two Intel Xeon processors. An Ethernet port is used to connect the simulator to a host PC. The host PC runs the RT-Lab software that is provided from the OPAL-RT company and it is used to control the real-time simulator. On the right side, we find a high performance FPGA that is connected with the real-time PC through a PCI-Express interface. This interface allows a fast communication and also a small latency in the transmission of the signals. The FPGA manages the external data acquisition and transmission through the I/O boards. A carrier board inside the chassis of OPAL-RT can host a maximum of 8 I/O modules that can be customized by the user to satisfy the requirements of the project. Four different module types are available:

- Analog-In module: it has 16 differential channels and it is used to acquire analog signals with a high sample rate of 2MSPS and a resolution of 16 bits;

- Analog-Out module: it has 16 channels and it is used to transmit analog signals in the range of $\pm 16V$;
- Digital-Input module: it has 32 channels and it is used to receive digital signals.
- Digital-Output module: it has 32 channels and it is used to transmit digital signals.

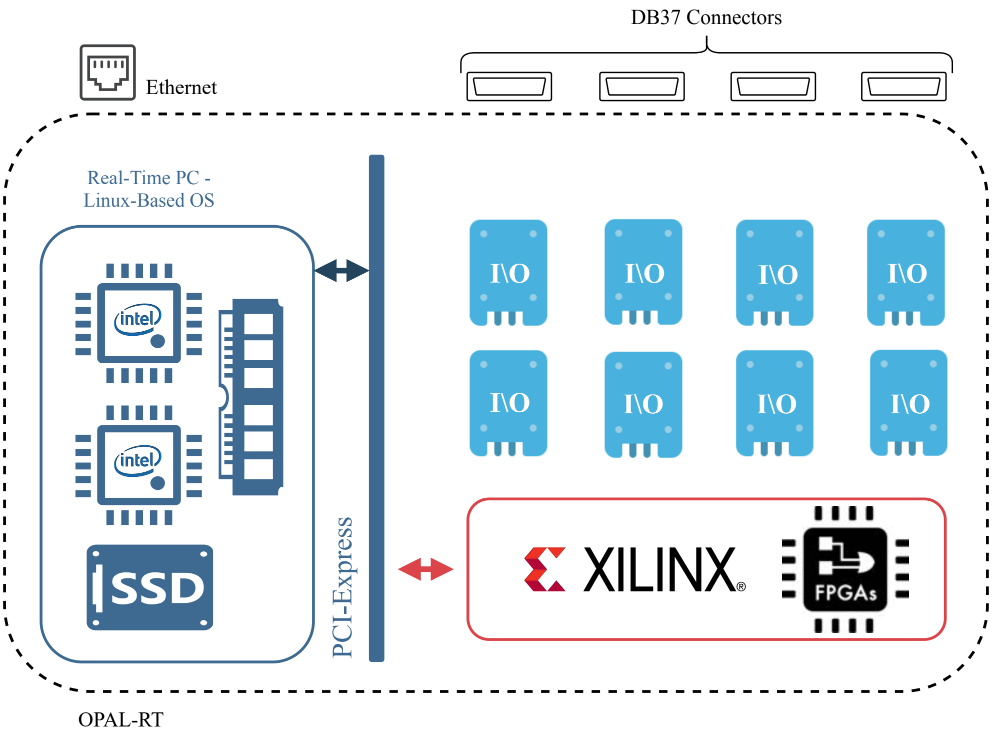


Figure 4.2: OPAL-RT architecture

The modules follows a proprietary form factor, called Type B mezzanines [29]. Each mezzanines can host an I/O module that is automatically detected at power-up. All the mezzanines are routed through the carrier board to the DB37 connectors in the back panel and at the same time to the RJ45 connectors in the front panel. The DB37 connectors are divided in 4 groups composed by 4 connectors labeled from 1 to 4. Each group has two sub-groups A and B composed of two DB37 connectors (P1 and P2). The group A is linked with the module present in the front mezzanine, the group B is linked with the module present in the rear mezzanine. Based on the module plugged in the mezzanine the connectors P1 and P2 will present a different configuration. If an Analog module that has 16 channels is connected, only the connector P1 carries the signals. If a Digital module that has 32 signals is plugged in the mezzanine, the first 16 channels are routed to the connector P1 and the rest to the connector P2.

Apart from the DB37 and RJ45 panels, the chassis has other several connectors as shown in Figure 4.3:

- (A, E) RJ45 and BNC monitoring interface;
- (B) SFP sockets;
- (C) Hardware synchronization connectors;
- (D) USB port for JTAG programming;
- (F) Standard computer connectors;
- (G) Bays for PCI cards;
- (H) DB37 I/O connectors;
- (I) 5V/12V power connector up to 4A;
- (J) GND screw;
- (K) Power plug and on/off switch;
- (L) Computer connectors;
- (M) Low-profile PCIe slots.

The DB37 connectors are designed to connect the control boards, the RJ45 connectors instead, are designed for a monitoring purpose. The user can directly plug an Ethernet cable from (A) to (B) and monitor the signals in real-time with an oscilloscope connecting a BNC cable to (C), as shown in Figure 4.4.

The model of the simulator used in this project is the OP5707. It has two Intel Xeon E5 8-cores CPUs, together with 4x8Gb of RAM and a Xilinx Virtex-7 FPGA. The simulator is equipped with an OP5332 Analog-out board with 16 outputs, an OP5342 Analog-in board with 16 inputs, an OP5353 Digital-in board with 32 inputs and an OP5360 Digital-out board with 32 outputs.

4.2.2 RT-Lab

RT-Lab is the software provided with the OP5707 real-time simulator. It is fully integrated with MATLAB/Simulink and allows to develop and simulate complex models with high reliability. The key feature of the software is its flexibility and scalability, which allows to gradually expand the model in a modular way in order to simulate very large systems. The software allows remote access and provides a customizable dashboard useful to interact with the real-time simulation and to represent the collected data.

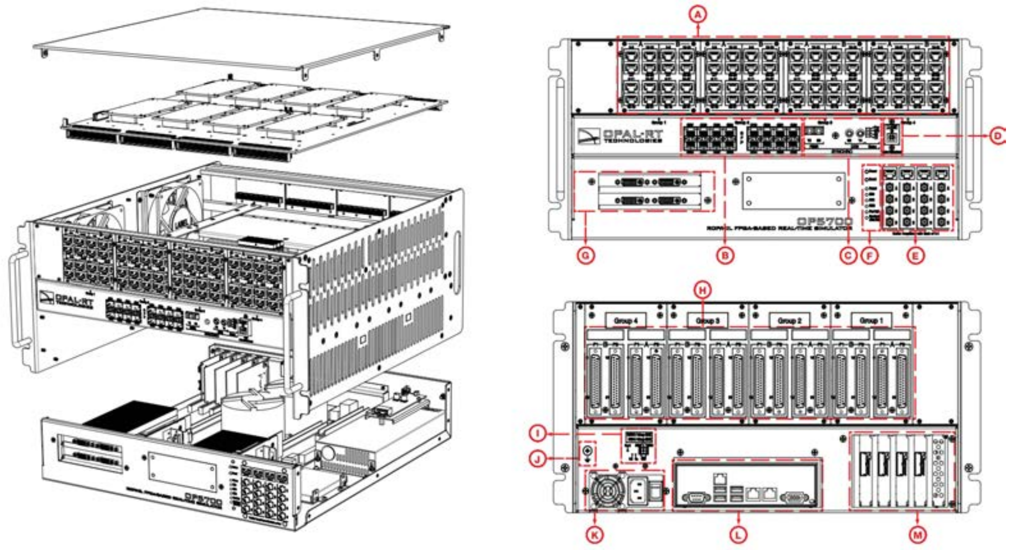


Figure 4.3: OPAL-RT architecture and front and back panels [30]

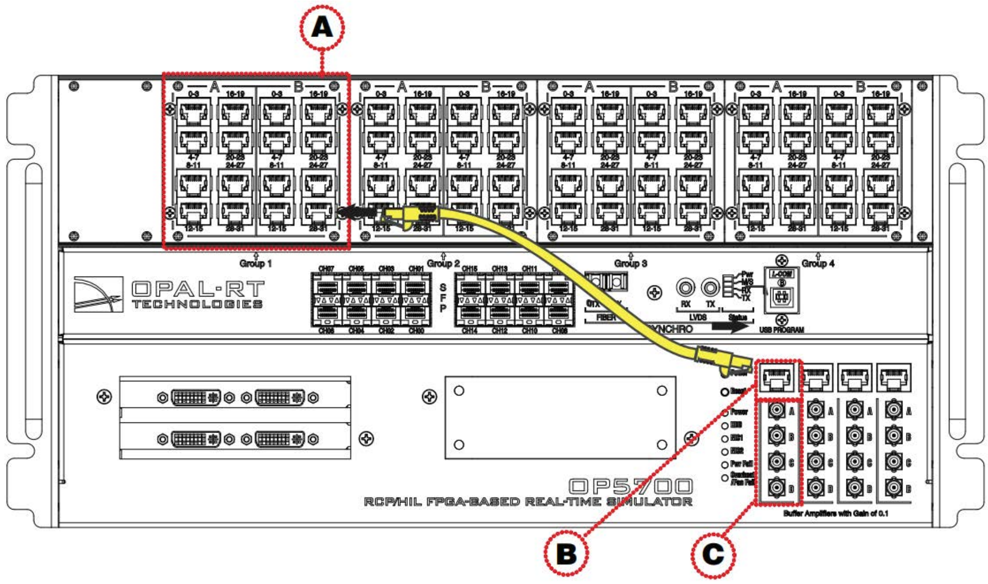


Figure 4.4: OPAL-RT RJ45 & BNC monitoring interface [30]

Our purpose is to run a CHIL simulation where multiple controllers and converters are employed and connected in a complex grid. RT-Lab handles the model editing, the code generation, and the data monitoring. In order to run the real-time simulation the procedure can be summarized in four main steps, as shown in Figure 4.5.

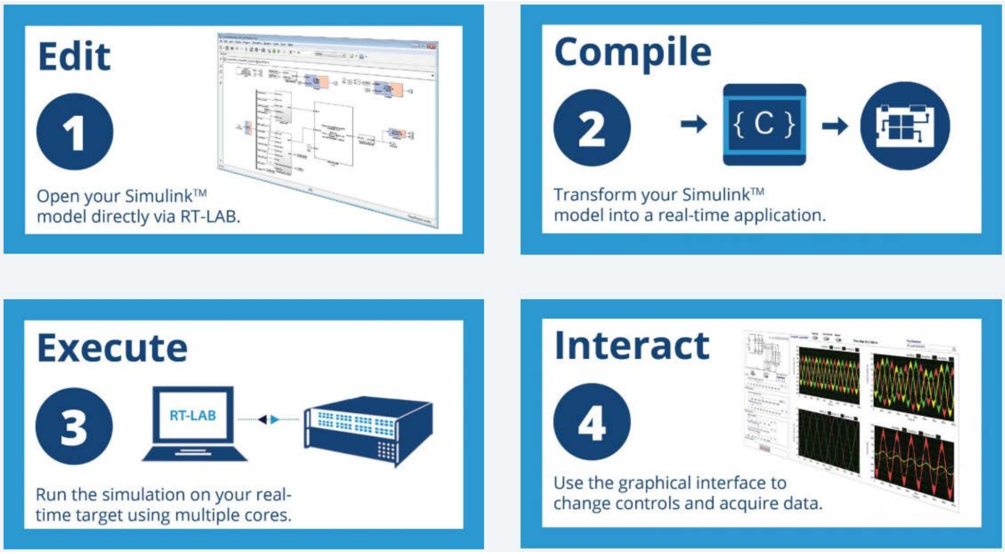


Figure 4.5: 4 Steps to Real-Time simulation [29]

It is supposed that the user is already familiar with Simulink as a modelling/simulation tool. Since we want to perform simulations of multiple converters controlled by the grid forming techniques the basic module to implement is the same as the one presented in Figure 3.11 in Chapter 3. RT-Lab uses Simscape Electrical™which is a library used for simulating power systems and power electronics in Simulink. It also provides a Simulink library which contains several blocks e.g. the eHS solver.

4.2.2.1 Model editing and building with the eHS solver

The first step is the editing of the model that can be done directly with Simulink opened via RT-LAB. For our purpose we use the eHS solver, which is a specific solver used for electric circuits provided by RT-Lab. To run a simulation, the user has to prepare two Simulink models, the circuit model and the RT-Lab model. The first one implements the power electronic components and the second one the controllers and the monitoring interface.

The circuit has to be designed using a special subset of blocks part ofthe SimPowerSystems library, which comprises Series/Parallel RLC Branch/Load, breaker, Linear transformers, mutual inductance, Voltage/Current measurement blocks, power switches and current/voltage source, etc. For a complete list of the supported block refer to the manual [29]. Component naming is very important and has to be done following this notation [29]:

- The sources should have the prefix "U" followed by a 2-digit index, starting from U01.
- The switches should be named with the prefix "SW" followed by a 2-digit index, starting from SW01.
- The measurements should have the prefix "Y" followed by a 2-digit index, starting from Y01.

In this way it is easy to map the components when the circuit is inserted in the RT-Lab model.

The RT-Lab model is split in two main blocks. The console block is used to monitor the data of the system, while the controller block contains the grid forming controllers used to control the inverters inside the circuit developed before. One of the fundamental blocks of the RT-Lab model is the eHS solver. According to the manual, the eHS block provides a sort of interface with the circuit model *"implementing the driver that manages all the communication with the eHs firmware including solver initialization and the transmission in real time of the circuit control signals"* [29]. In other words it acts like a portal that allows us to send/receive current and voltage source control signals, switching commands to the transistors, and voltage and current measurements, as shown in Figure 4.6. The eHS block is needed because the power electronic components are implemented inside the FPGA, while the grid forming controllers and the data monitoring are managed by the CPU. This part will be explained in details in the next step.

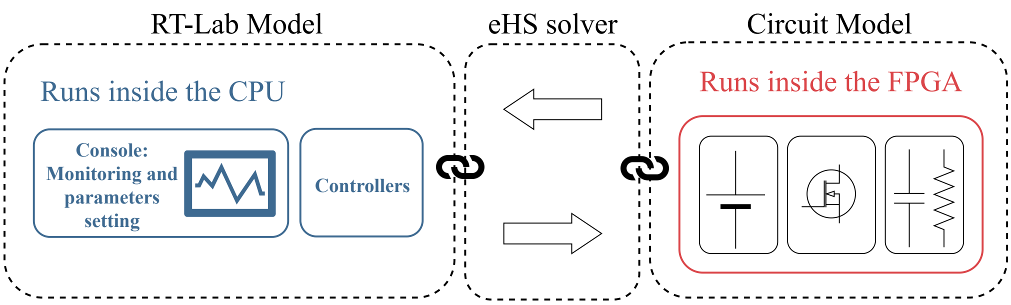


Figure 4.6: Rt-Lab and Circuit model

So, to sum up, the user has to:

- Prepare a model of the circuit with all the power components and label the block correctly as described before.
- Prepare the RT-Lab model that is composed of two main parts. The console part used to monitor the data, and the controller part where the controllers and the eHS block are implemented.

4.2.2.2 Compiling the system model

The building and loading of the models are automatic procedures managed by the RT-Lab software. It is very important to understand what the compiler does and how the two models developed in the previous step are treated.

As mentioned before, the OPAL-RT has inside a CPU and an FPGA, that have different tasks. The FPGA is very fast and powerful and for that reason it controls the communication with the I/O modules acquiring and receiving external signals, but also it simulates the power hardware. The model built inside the FPGA can be updated every 200 ns, which is perfect to emulate power electronics. The CPU instead, runs the grid forming controllers and collects the data coming from the FPGA through a PCI-Express interface. The refresh rate of the RT-Lab model can be much slower then the refresh rate of the circuit model, i.e. around 20-40ms. Due to this particular CPU-FPGA architecture, OPAL-RT is able to simulate complex systems with multiple converters connected to a complex electrical network. Splitting the model in multiple parts and updating them at different frequencies, reduce considerably the computational power necessary to run a CHIL simulation. The FPGA, which is particularly adapted to compute simple mathematical operations with high efficiency, is used to simulate the power hardware ensuring a high refresh rate. The controllers instead require complex mathematical operations but a lower refresh rate. they can be executed by the CPU. This is the crucial point that distinguishes OPAL-RT from the other real-time simulators allowing to run CHIL simulations with a high accuracy.

During the building process the compiler proceeds in several steps that can be summarized as shown in Figure 4.7.

Step 1: The compiler produces a Simulink model called "Console.slx". It represents the console interface which allows the user to interact with the real-time simulation, monitoring the data and at the same time to charge the parameters of the model. The "Console.slx" runs in the host PC and communicates with the simulator via a Ethernet port.

Step 2: The compiler automatically generates the c-code that implements the grid forming controllers developed in the RT-Lab model and loads them inside the real-time simulator to be run in the CPU. The CPU also takes care about the PCI-Express communication with the FPGA and the data exchange with the host PC.

Step 3: In the last step the eHS solver translates the circuit model producing a bit-stream file and the circuit net-list. The bit-stream is used to program the FPGA. During the translation task the eHS block automatically establish the communication channels with the CPU needed to link the circuit with the controllers, i.e. to link the circuit model with the RT-Lab model.

It is important to mark that the FPGA and the CPU are synchronized together, allowing a fast communication with no data losses. The real-time simulator is based on a hierarchical structure. At the top of the pyramid is the Host-PC which sends the data to the CPU of the real time simulator, (which sits at the

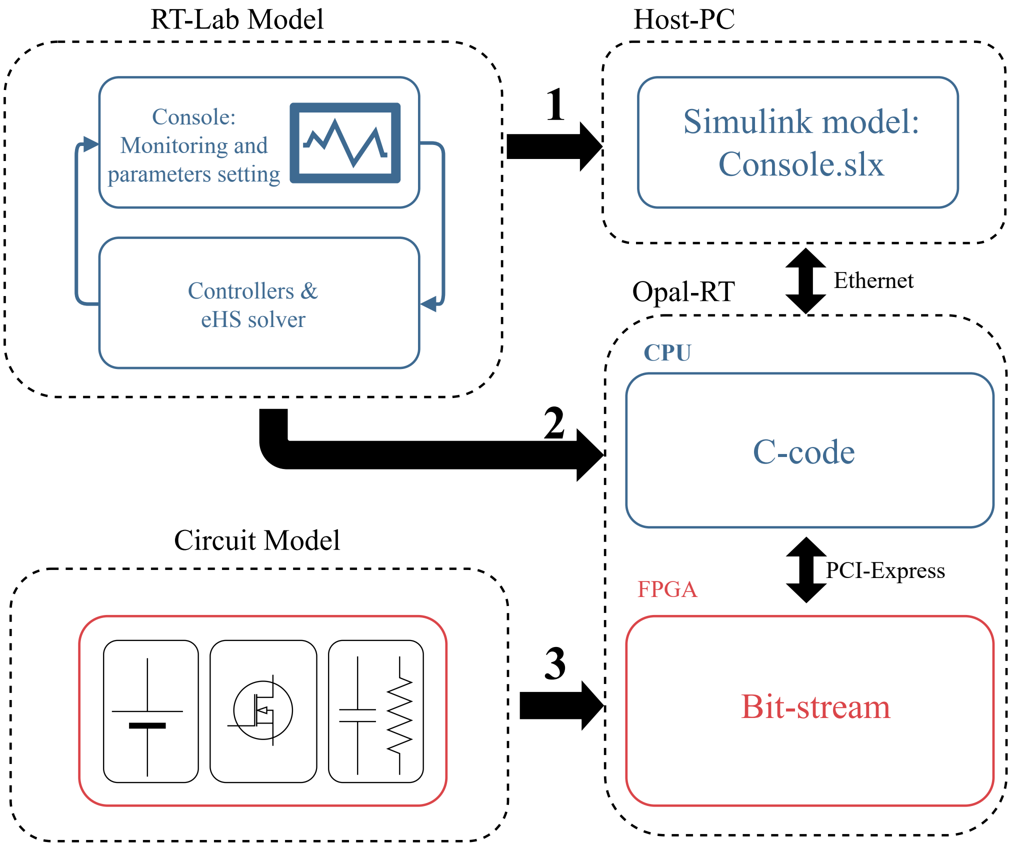


Figure 4.7: Compiler building steps

second level of the hierarchy). Here the data is processed and sent to the FPGA (third layer), forming a chain Host-PC→OPAL-RT CPU→OPAL-RT FPGA. In the same way following the chain in the opposite direction the data is sent from the FPGA to the Host-PC passing through the CPU and closing the loop. To exchange signals between the Host PC and the CPU of OPAL-RT a particular block called *OpComm* is used. The RT-Lab model has an *OpComm* block in the console sub-model to receive the data from the CPU and an *OpComm* block in the *Controller* sub-model to send the data to the Host-PC. It is important to specify that the *OpComm* block is not only used for communication between the host PC and the CPU, but also between sub-models running on different cores of the CPU. To exchange data between the FPGA and the CPU of OPAL-RT there are dedicated blocks called *Datain* and *DataOut* that the user can configure and customize. In our case we use them through the eHS block that already integrates them. A representation of the data exchange is shown in Figure 4.8.

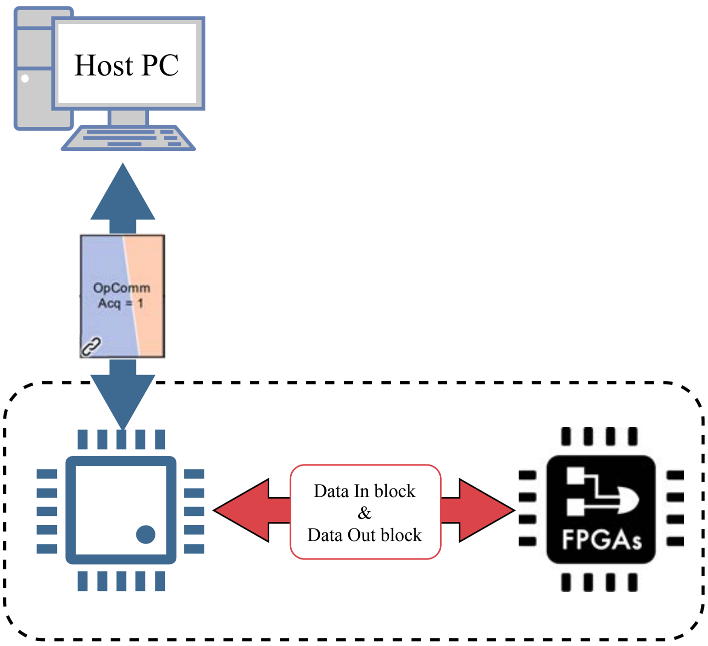


Figure 4.8: Data exchange architecture of OPAL-RT

Execute the real-time simulation Once the model is built and loaded to OPAL-RT, the user can run the simulation using the RT-Lab software. The process is straightforward because the RT-Lab automatically opens the "Console.slx" file and starts the simulator. At this point the user is able to monitor the data interacting with the "Console.slx".

Interact with the real-time simulation It is possible to interact with the "Console.slx" model in real time, changing parameters that for example trigger a load step or disconnect a converter in the circuit model, but also recording data in files. The scopes can be used as a sort of oscilloscope for monitoring the real time signals.

4.3 CHIL simulation for system level validation

The idea is to extend the model used for the Component Level Validation, connecting multiple grid forming converters with a complex electrical grid. As explain before, the circuit model is coded inside the FPGA of OPAL-RT, for the grid forming controllers instead there are three possible configurations.

Controllers and power electronics executed by OPAL-RT OPAL-RT can be used to simulate both controllers and power electronics. A schematic representation is shown in Figure 4.9

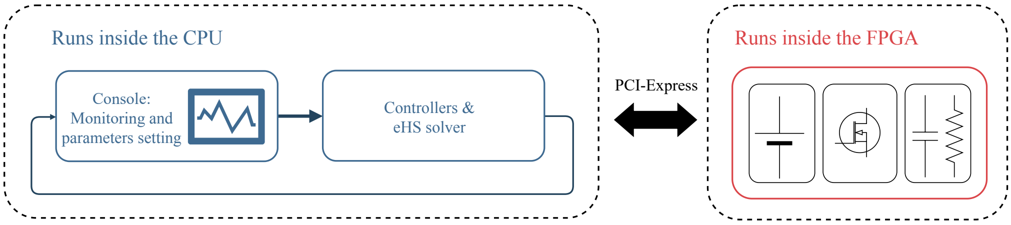


Figure 4.9: Controllers and circuit model executed by the OPAL-RT

The controllers run inside the OPAL-RT CPU while the power electronics inside the FPGA exchanging data via the PCI-Express interface which emulates the "physical connections" between the two systems. This approach can be used for the initial tests in order to study the behaviour of the controllers introducing delays in the signal transmission. In this scenario all the physical connections are missing and with that all the problems related to noise and physical limitations of the IO interfaces.

Controllers executed by control cards and the power electronics executed by the OPAL-RT A second possible configuration is presented in Figure 4.10.

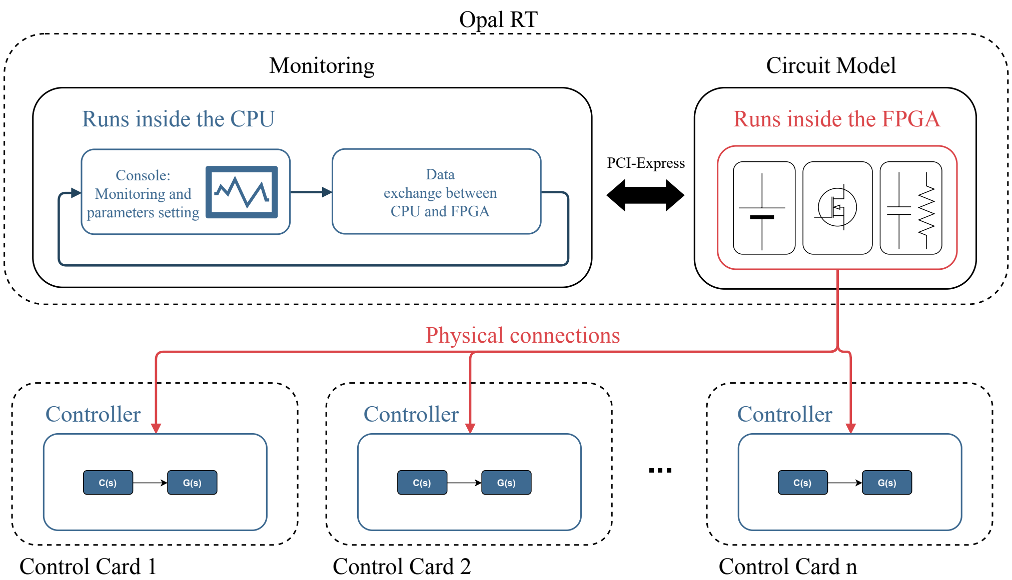


Figure 4.10: Controller executed by the control card and power electronics executed by the OPAL-RT

The setup of Figure 4.10 is more realistic. The power electronics are emulated inside the FPGA of OPAL-RT, while the CPU takes care only about the communication with the Host PC. The eHS block is configured so that the MOSFET gate control signals and the measurements are sent and received by the FPGA through the Analog and Digital I/O modules. In the previous configuration, the control signals were exchanged between the CPU and the FPGA through a PCI-Express interface. As explained before, the I/O modules are routed to the back panel with DB37 connectors where the user can plug multiple control boards. With this scenario we are able to reach the same accuracy obtained in the Chapter 3, but performing system level validation. This gives access to an area that has not been explored yet by the scientific community, and lays the foundation for testing more complex models in order to give an answer to all questions introduced at the beginning of the chapter.

Hybrid CHIL simulation The last scenario presents a hybrid configuration between the two explained before. For example, imagine a system composed of multiple converters and the goal is to test only one particular controller. The idea is to implement the specific controller inside a control board and the remaining controllers inside the CPU of OPAL-RT producing a hybrid system. The same solution can be applied when we have multiple converters and synchronous machines. The latter can be implemented inside the CPU, while the controllers are executed on the control boards. This is possible thanks to the flexibility and the characteristics of OPAL-RT and its architecture. In particular to the fact that different parts of the model run at different refresh rates.

4.3.1 The nine-bus system model

To validate the grid forming techniques at a system level a more complex model than the one presented before is required. A suitable model is shown in Figure 4.11 and it is a modified version of the IEEE nine-bus system.

The nine-bus system is composed of three converters that operate in the low voltage range (1KV). Each converter is an aggregation model of 200 AC coupled converters with a nominal power of 500kVA each, producing a total power of 100MVA. The three converters are connected to the 13.8kV buses through the LV/MV transformers (low voltage/ medium voltage) and in the same way to the 230kV buses through the MV/HV transformers (medium Voltage/ high voltage). The transmission lines are modeled by three-phases RL components. The system presents four constant impedance loads, where one of them can be connected and disconnected to the network through a breaker. The nine-bus system is suitable to perform system level validation of the grid forming techniques because it incorporates multiple converters connected together. It gives the possibility to perform real-time simulations with multiple scenarios. For example, connecting and discounting different loads to the network or emulating the disconnection of a synchronous generator. The nine-bus system will be used to perform a CHIL

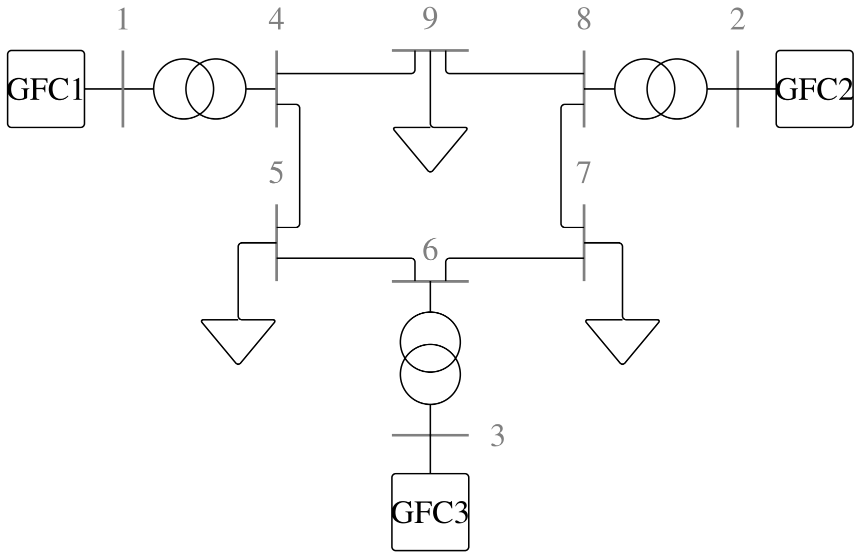


Figure 4.11: The nine-bus system model

simulation at the system level, where the controllers and the power electronic will be executed by OPAL-RT following the steps presented in Section 4.2.2. First of all we build the circuit model composed of the power electronics components, as shown in Figure 4.12. Here, each converter is realized as shown in Figure 4.13. This model will be translated to VHDL code by the eHS solver to be executed inside the FPGA. At the same time, the eHS solver will take care of establishing all the communications with the CPU to send the measurements to the controllers and to receive the values of the voltage sources ("U01Vdc", "U02Vdc", "U03Vdc") as well as the signals to control the MOSFETs.

In the second step we build the RT-Lab model that is composed of two main blocks, the *Console block*, and the *Controller block*. They are connected together forming a loop. The *Controller block* receives the data from the *Console block* in order to perform a load step, and at the same time it sends the voltages and the currents measurements of the converters to the console for monitoring purposes.

Inside the *Controller block* there are the grid forming controllers and the eHS solver, as shown in the Figure 4.15. The three controllers GFC1, GFC2, GFC3, receive the current and voltage measurements from the eHS block that acts like a bridge to the circuit model that runs inside the FPGA. Each controller produces as output the modulation signals. They are converted by three PWM Generator blocks to PWM signals that control the MOSFETs gates of the inverters inside the circuit model. The eHS solver receives as input these PWM signals in the GATES RTE port and the three voltage references to control the dc side of the converters. Finally all the measurements are sent to the Console block through the "out8" port.

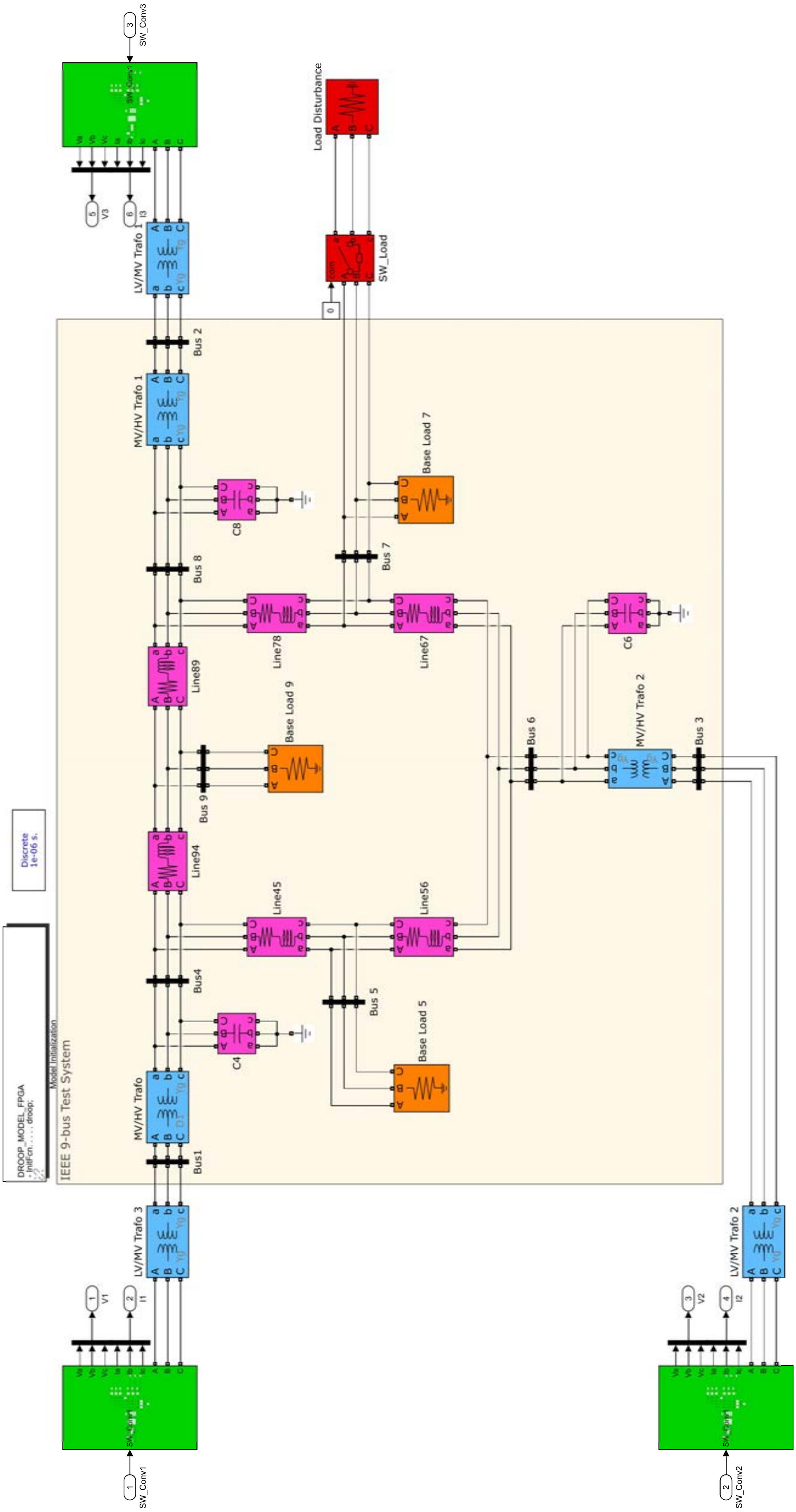


Figure 4.12: Network model of the nine-bus system

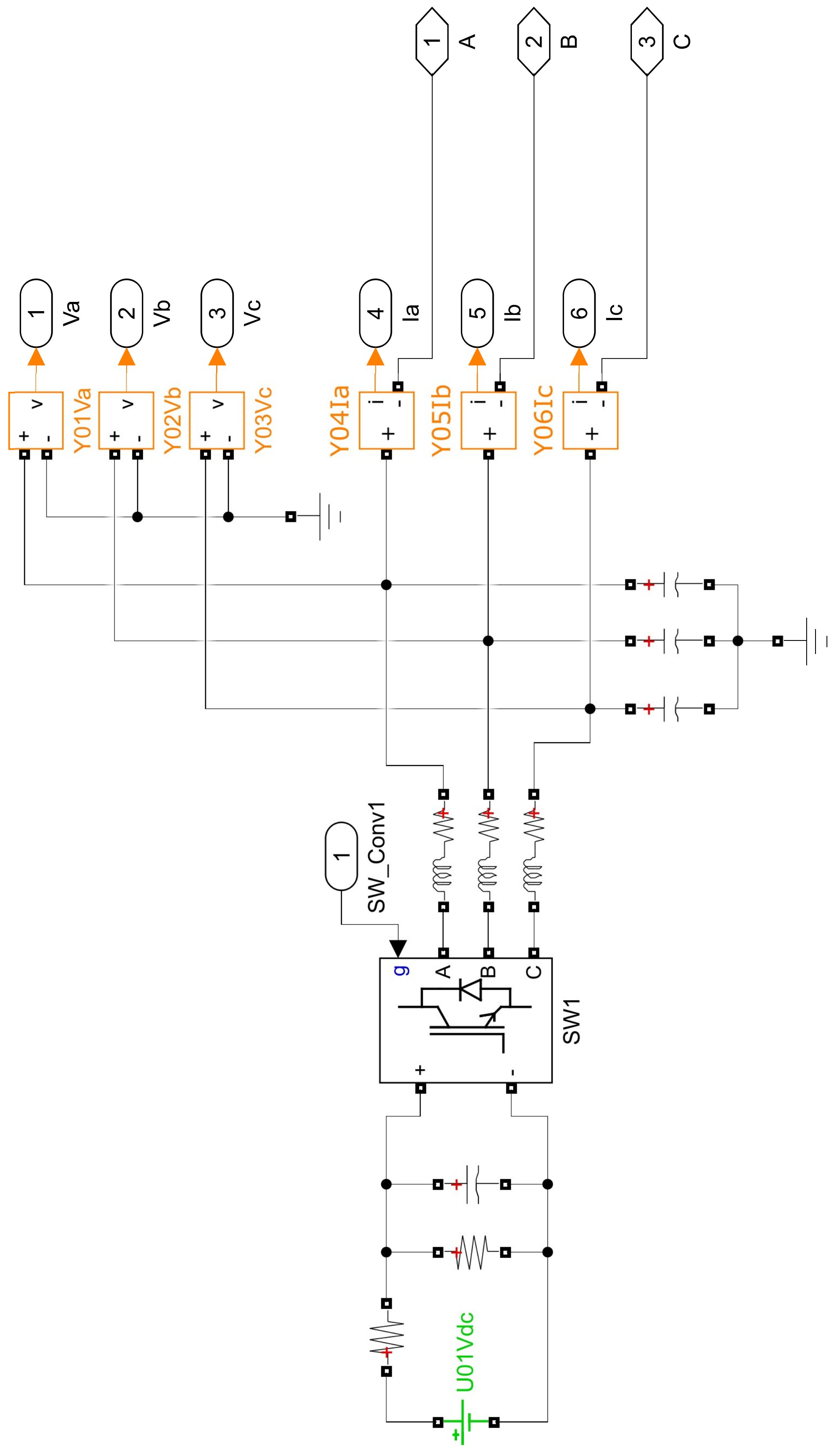


Figure 4.13: Power electronics of each converter of the nine-bus system

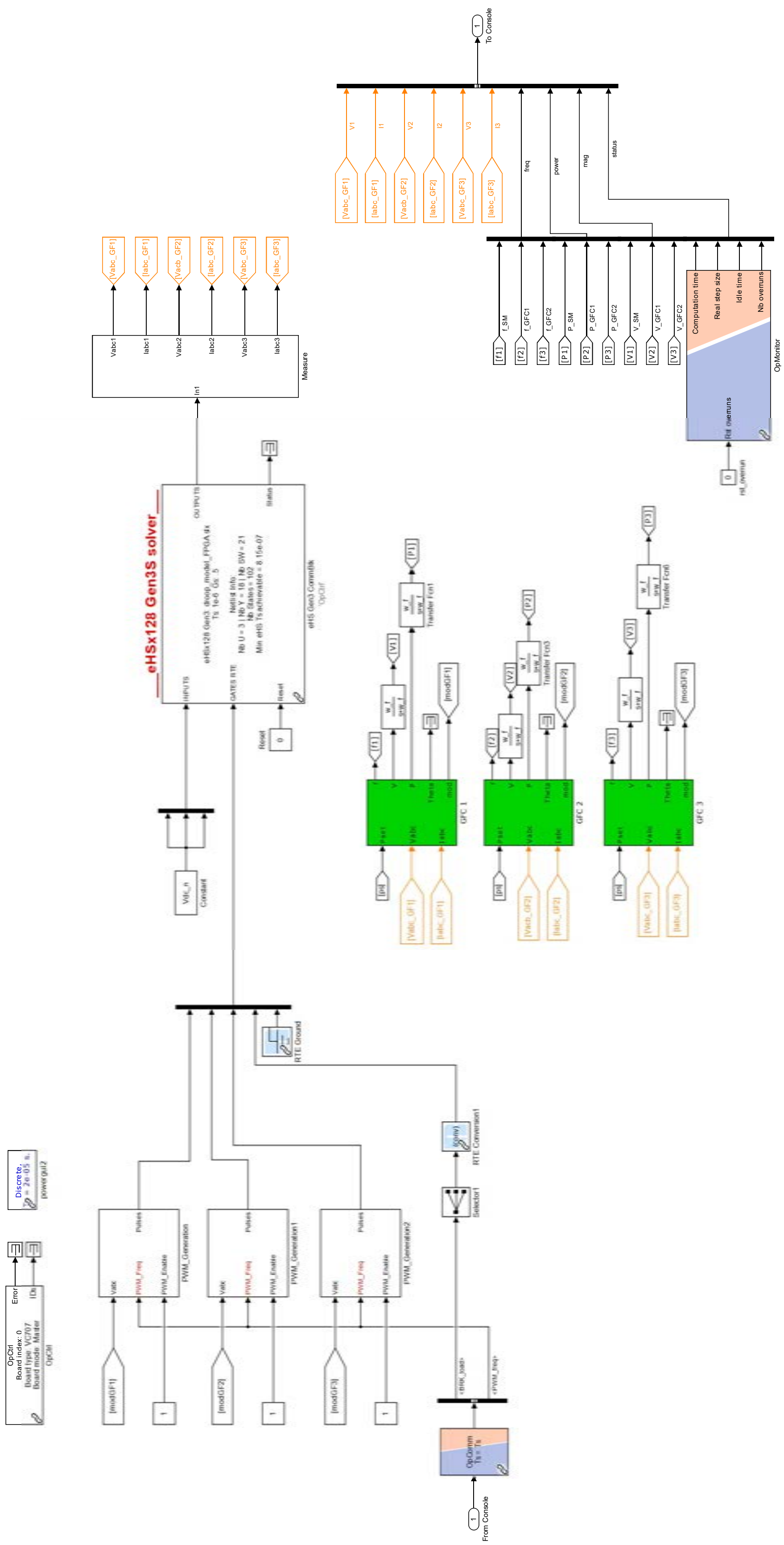


Figure 4.14: RT-Lab model: Controller block which contains the implementation of the grid forming controllers and the eHS block.

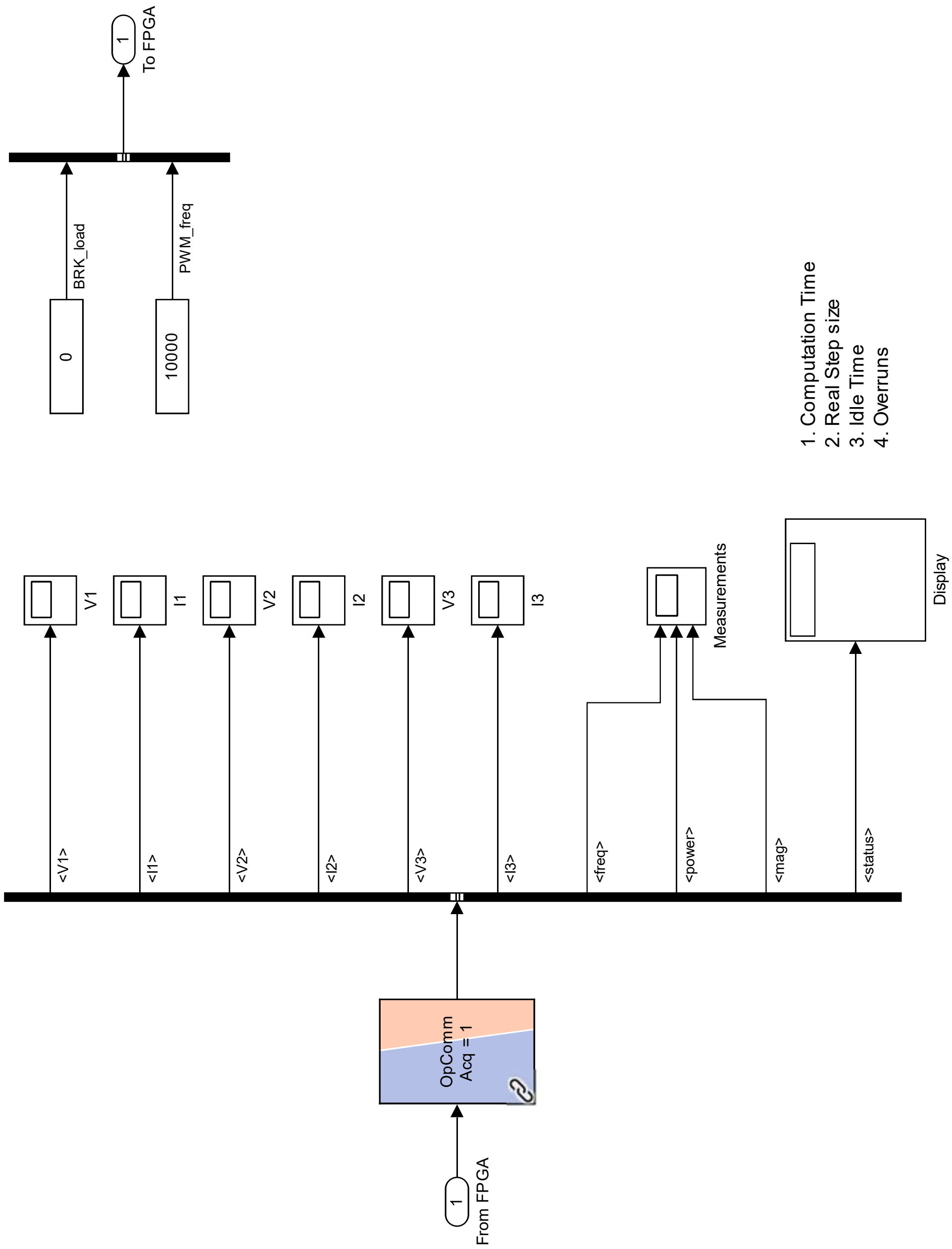


Figure 4.15: RT-Lab model: Console block

The *Console block* is used to monitor the data and it is shown in Figure 4.15, Here, all the measurements can be monitored in real time using the *Scopes blocks* that act like an oscilloscope. Furthermore, the user can set the *BRK_ load block* to "1" to trigger a load step. This closes the breaker highlighted in Figure

4.12, which connects the load to the grid. On the user side the procedure is completed and now the RT-Lab software automatically compiles and loads the models inside the OPAL-RT.

During the building phase the compiler splits the RT-Lab model in two parts producing two different Simulink file, to represent the *Console block* and the *Controller block*. The *Console block* during the real time simulation runs in the host PC and can be use to monitor the data, the *Controller model* instead runs inside the CPU of the real-time simulator.

To sum up, our system is composed of:

- The circuit model that runs inside the FPGA of the simulator and represents the power electronics. The compiler automatically produces a bit-stream file to code the FPGA of OPAL-RT.
- The model that represents the *Controller block* that runs inside the CPU of the simulator and represents the grid forming controllers. Moreover it collects all the measurements from the FPGA and sends them via an Ethernet interface to the host PC. In the same way it receives the signal to perform the load step from the host PC and sends it to the FPGA.
- The model that represents the *Console block* runs inside the host PC and is used to monitor the data and interact with the system.

Running a simulation og OPAL-RT produces the results shown in Figure 4.16. The system is perturbed by connecting a constant impedance load, which consumes $0.75[pu]$ of the nominal power of the system. The graphs show the frequency, the power and the voltage magnitude of the three converters. The system behaviour is very close to the one obtained with the offline simulations. All three converters respond adequately to the load change keeping the frequency stable, increasing the power provided to the system and tracking the voltage reference. The results obtained running the offline simulation are presented in Figure 4.17. In general they present less noise compared to the measurement obtained with the CHIL simulation. An important difference appears in the initial frequency: for the real-time simulations for all the converters the frequency is approximately $49.6[Hz]$ while in the offline simulation it is approximately $49.9[Hz]$. At the moment we do not have an explanation for this behaviour. More simulations and analysis would be required, but this is left as future work.

This setup can be used as a first approach towards performing a CHIL simulation at the system level. Mainly, it can be used by the user to familiarize himself with the OPAL-RT and to test if the circuit and RT-Lab model work properly. Generally, a more realistic scenario is required to guarantee the stability of the network and to have the same accuracy level used to perform component level validation, as shown in Chapter 3. For that reason it is necessary to move the controllers outside the simulator and implement them inside multiple control boards. To build a setup with this configuration, i.e. the same as explained in

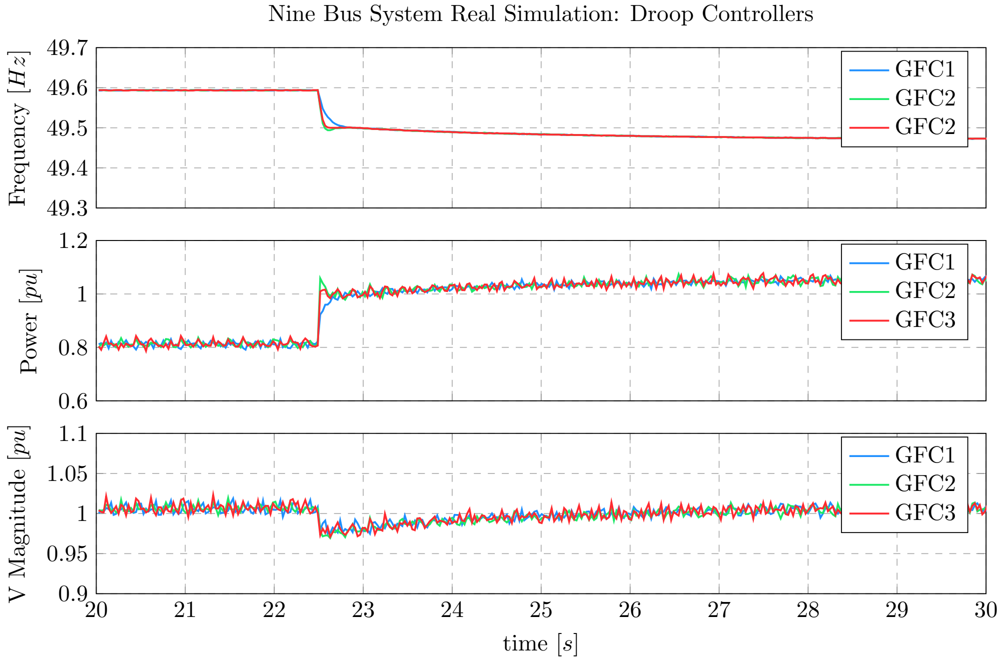


Figure 4.16: Real-time simulation of the nine-bus system using the droop control strategy for the grid forming controllers.

the Figure 4.10, an interface board is required. Since the companies that produce the real-time simulator and the control card do not provide any suitable solution we developed developing a custom one as part of the work performed for this thesis.

4.4 Interface board for connecting OPAL-RT with multiple control cards

The voltage levels of the IO of OPAL-RT are not directly compatible with the control board so for that reason it is necessary to develop a custom interface able to mach the specification of both systems. The idea is to build a platform that guarantees good performance, with a better flexibility compare to the old setup, and that is more user friendly.

Since all the features of the F28379D chip are not implemented in the LaunchPad board we decided to use a custom control card developed internally at AIT (Austrian Institute Of Technology GmbH).

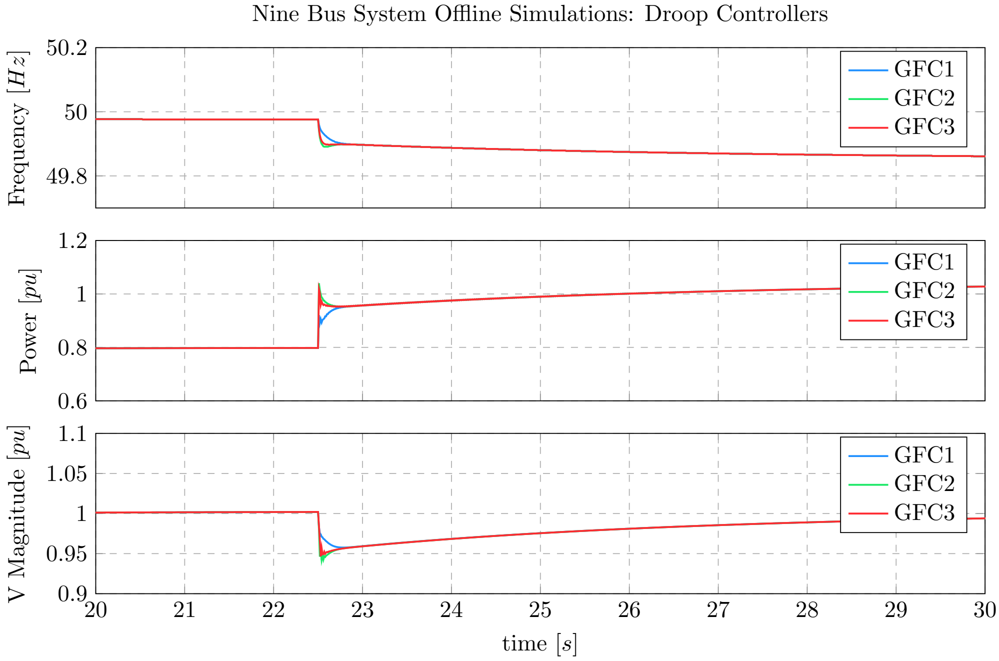


Figure 4.17: Offline simulation of the nine-bus system using the droop control strategy for the prior forming controllers.

4.4.1 Custom control board based on the F28379D chip

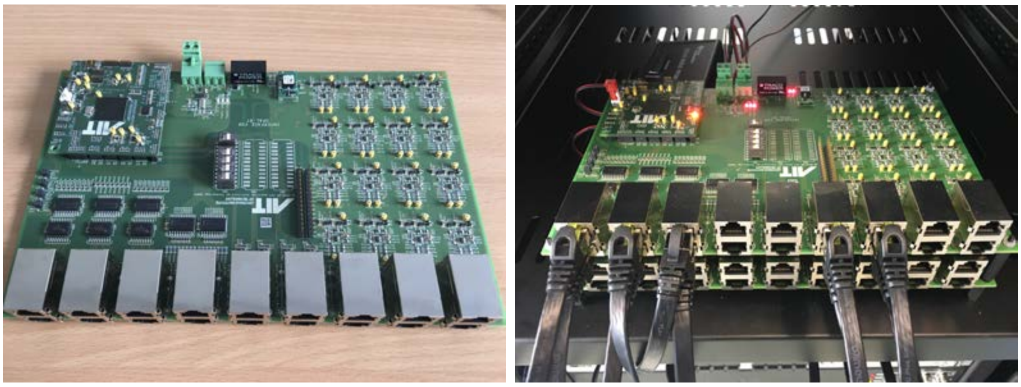
The new control board is based on the F28379D chip that is also present in the LaunchPad. Therefore, the code developed for the old platform still remains compatible with the new one. In addition it is possible to benefit from new features. Compared to the Launch-Pad it has more Digital PWM capable outputs, but also provides two Serial Communication ports (SCI-A, SCI-B) and two Serial Peripheral Interface ports (SPI-A, SPA-B). The new control board does not have an on-board debug probe so in order to program and debug the TI embedded processor we need an external one. The one available in the laboratory is the model "Texas Instruments XDS110 Debug Probe".

4.4.2 The OPAL-RT/F28379D interface board

We developed the OPAL-RT/F28379D interface board to be:

- **Flexible:** in many occasion it is very useful to change the signal configuration in order to match the project requirements. Having the possibility to route the signals in the desired way is a crucial point in terms of the setup flexibility. The interface board present several RJ45 connectors that provides access to Analog and Digital IOs. In this way the user can easy decide how to route the signals from the control board to OPAL-RT by just plugging an Ethernet cable in the right place. Since the F28379D chip doesn't support directly analog outputs, the interface board compensates the missing feature introducing high-resolution DACs.

- **Modular:** since to perform system level validation the setup requires the interconnection of multiple converters, it is preferable to have multiple control cards where each of them controls a converter. To satisfy this requirement, multiple interface boards can be stacked together, as shown in Figure 4.18(b).
- **User friendly:** thanks to the RJ45 connectors, the interface board presents an user friendly interface. It requires common Ethernet cables to connect it to the OPAL-RT. In addition it presents several test points (the small yellow dots visible in Figure 4.18) to monitor the IO signals and several DIP switches used to enable or disable the DACs.



((a)) ((b))

Figure 4.18: The image on the left shows the developed interface board, the image on the right two multiple interface boards stacked together.

A list of the features that we included in the interface board features is reporter in Table 4.1.

Let's now analyze the principal part of the circuit to understand how it works and how it guarantees the signals compatibility between the control card and OPAL-RT.

Power-Supply: the interface board can be directly supplied with OPAL-RT, which has a small connector in the rear panel. The connector provides two main voltages, 5V and 12V, rated at 4A maximum each. The interface board has a maximum power consumption of 0.5A so it is possible to supply several of

Features	Quantity
Digital Inputs	16
Digital Output	24
Analog Inputs	16
Analog Outputs	16
SPI port	2
SCI port	3
DIP switches	16
Analog In Test points	16
Digital In Test points	16

Table 4.1: Interface board features

them with the power supply of OPAL-RT. The interface board has two plug and play connectors that receive the 5V and 12V supply voltage. This supply is also used to power the control card. In this way no other external power supplies are required and the setup (OPAL-RT+Interface board+Control board) can work independently.

Digital Inputs to OPAL-RT The Digital Inputs to OPAL-RT support a voltage range of 4V-50V. This is not compatible with the range of the control card, i.e. 0V-3.3V. To make them compatible we use the *ULN2803A* integrated circuit (IC) from Texas Instruments [31]. The IC is a Darlington transistor array that can be used specially as a logic buffer; in our case it receives as input a signal in the 0-3.3V voltage level and produces as output a signal that has a voltage level of 0-12V, which is compatibles with the OPAL-RT specifications. The IC works like an inverting logic buffer, i.e. the output signal is inverted with respect to the input signal. Hence, another IC is used to negate again the signal, the *MC74AC541* [32] that is an inverting buffer. It is placed before the ULN2802A in this way thanks to the double negation the output signal from the interface card has the correct voltage level. A graphical representation of the signal conversion chain is shown in Figure 4.19.

Digital Outputs from OPAL-RT: The Digital Outputs from OPAL-RT have a range of 5V-30V. The voltage level of the Digital Outputs can be selected by the user. The DB37 connectors dedicated to the the Digital Outputs, in the rear panel of OPAL-RT have two specific pins where the user has to provide the voltage reference for the output signals. In our case we used a voltage level of 12V because it is already available directly from OPAL-RT (we used same the 12V used that supply the interface board). The Digital Input to the control card tolerates a signals in the voltage range of 0-3.3V. So to guarantee the compatibility with the Digital Outputs from OPAL-RT a voltage divider circuit is used. We added also a non-inverting buffer after the voltage divider (MC74AC540). A voltage buffer circuit is used to transfer a voltage from a first circuit with an

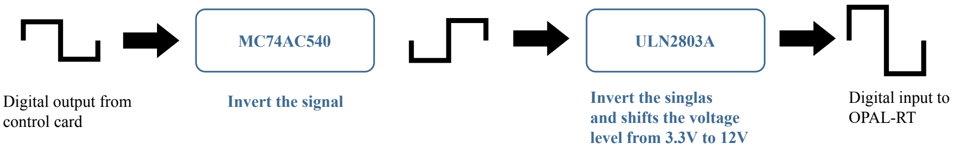


Figure 4.19: The input signal is inverted by the MC74AC541 and then inverted again by the ULN2803A IC that also shifts the voltage level of the input signal from 3.3V to 12V.

high output impedance to a second circuit with a lower input impedance. The buffer avoids that the second circuit overcharges the first circuit. In other words it is used as a protection circuit.

Analog Inputs to OPAL-RT: Since the control card does not support directly analog output signals, we integrated DACs in the interface board. There are many types of DACs developing on the communication protocol used to receive the data and the resolution of the output signal. Most of them work with the SPI or I^2C protocol, or receiving as input a PWM signal. For this project we selected the last type because it does not require further software configurations and all the PWM signals are already available as outputs from the control card. The PWM signal produced by the control card can be used to control the DACs or as digital output. To select how to route the PWM signals, the interface board integrates DIP switches used to route the PWW signals to the Digital In to OPAL-RT RJ45 connectors or to the Analog In to OPAL-RT RJ45 connectors.

The DAC IC used is the model **LTC2645** [33] and it has four channels. It is capable to receive as input four PWM signals in the 30Hz-100KHz frequency range and to produce four analog signals in the voltage range of 0-5V. Based on the frequency of the PWM input signal to LTC2645, the resolution of the analog output signal produces by the chip, change. For example, if the input PWM signal has a frequency of 6.25KHz the analog output has a resolution of 12-bit, if it has a frequency of 100KHz the analog output has a resolution of 8-bit . In our project the input PWM signal frequency is 10KHz. This guarantees a resolution of 10 bit. The outputs signals from the DACs are compatible with the analog input voltage range of OPAL-RT that is ± 20V, so no other components are required.

Analog Outputs from OPAL-RT: The Analog Outputs from OPAL-RT have a voltage range of $\pm 16V$. The control card instead can receive analog signals in the range of 0-3V. Hence it is necessary to compress the analog outputs from OPAL-RT to match the control card voltage limits. One solution is to use operational amplifiers. The circuit used to achieve our goal is shown in Figure 4.20 and the list of components used to build the circuit is shown in Table 4.2.

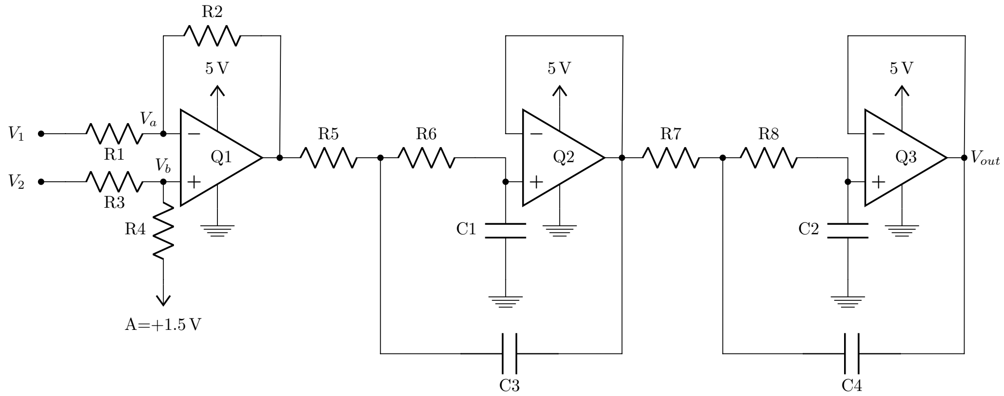


Figure 4.20: Analog circuit used to compress the Analog Outputs from OPAL-RT in the 0-3V range in order to respect the limits of the control board.

IC name	Value	IC name	Value
R1	100KΩ	C1	11ρF
R2	3.3KΩ	C2	10ρF
R3	100KΩ	C3	100ρF
R4	3.3KΩ	C4	100ρF
R5	13.3KΩ	Q1	LTC6252
R6	48.7KΩ	Q1	LTC6252
R7	42.2KΩ	Q2	ADA4891
R8	140KΩ	Q3	ADA4891

Table 4.2: List of components used to build the analog input interface.

The LTC6252 IC [34] is an operational amplifier and it is used in a differential configuration to compress an input signal in the range of ±16V to a signal in the range of 0-3V.

Let us now compute the gain of the amplifier shown in Figure 4.20. Consider V_1 and V_2 measured taking the point A as the zero volt reference. V_b is computed with 4.1.

$$V_b = V2\left(\frac{R4}{R4 + R3}\right) \tag{4.1}$$

If $V_2 = 0$, then we obtain (4.2) and if $V_1 = 0$ we obtain (4.3).

$$V_{out(a)} = -V_1 \left(\frac{R2}{R1}\right) \tag{4.2}$$

$$V_{out(b)} = V_2 \left(\frac{R4}{R4 + R3}\right) \left(\frac{R1 + R2}{R1}\right) \tag{4.3}$$

V_{out} is the sum of $V_{out(a)}$ and $V_{out(b)}$ as shown in (4.4) and substituting (4.2) and (4.3) in (4.4) we obtain the relation shown in (4.5).

$$V_{out} = V_{out(a)} + V_{out(b)} \tag{4.4}$$

$$V_{out} = -V_1 \left(\frac{R2}{R1}\right) + -V_2 \left(\frac{R4}{R4 + R3}\right) \left(\frac{R1 + R2}{R1}\right). \tag{4.5}$$

Supposing $R1 = R3$ and $R2 = R4$ (4.5) becomes (4.7).

$$V_{\text{out}} = \frac{R2}{R1}(V_2 - V_1) \tag{4.6}$$

At the end when, changing the reference point A to GND an offset of 1.5V must be added to V_{out} of (4.6),thus, obtaining the finial equation (4.7).

$$V_{\text{out}} = \frac{R2}{R1}(V_2 - V_1) + 1.5V \tag{4.7}$$

In addition, the circuit is composed of two ADA4891 operational amplifiers that are used as a low pass filter. The cut off frequency of the low pass filter is set to 1MHz.

4.4.3 Simulation of the main circuit functional blocks

To verify if the circuit works it is a good practice to simulate it with a circuit simulator such as LTspice. This software allows to perform precise simulation of analog circuits before manufacturing the PCB. It is possible to monitor the voltages and currents of any single node in order to understand clearly how the circuit behaves. In our case we simulated the circuit that receives the analog signals from OPAL-RT. LTspice produces the results shown in Figure 4.21. The input signal is a sine wave with an amplitude of 16V, a frequency of 50Hz and a zero DC offset. The output of the circuit in Figure 4.21 is a sine wave with the same frequency but with an amplitude of 2.55V and a DC offset of 1.5V that stays inside the limits of the control card. Once the main functional blocks of the interface board were developed it is possible to produce a schematic that represents all the connections between the electronic components and after that the PCB design for manufacturing the board.

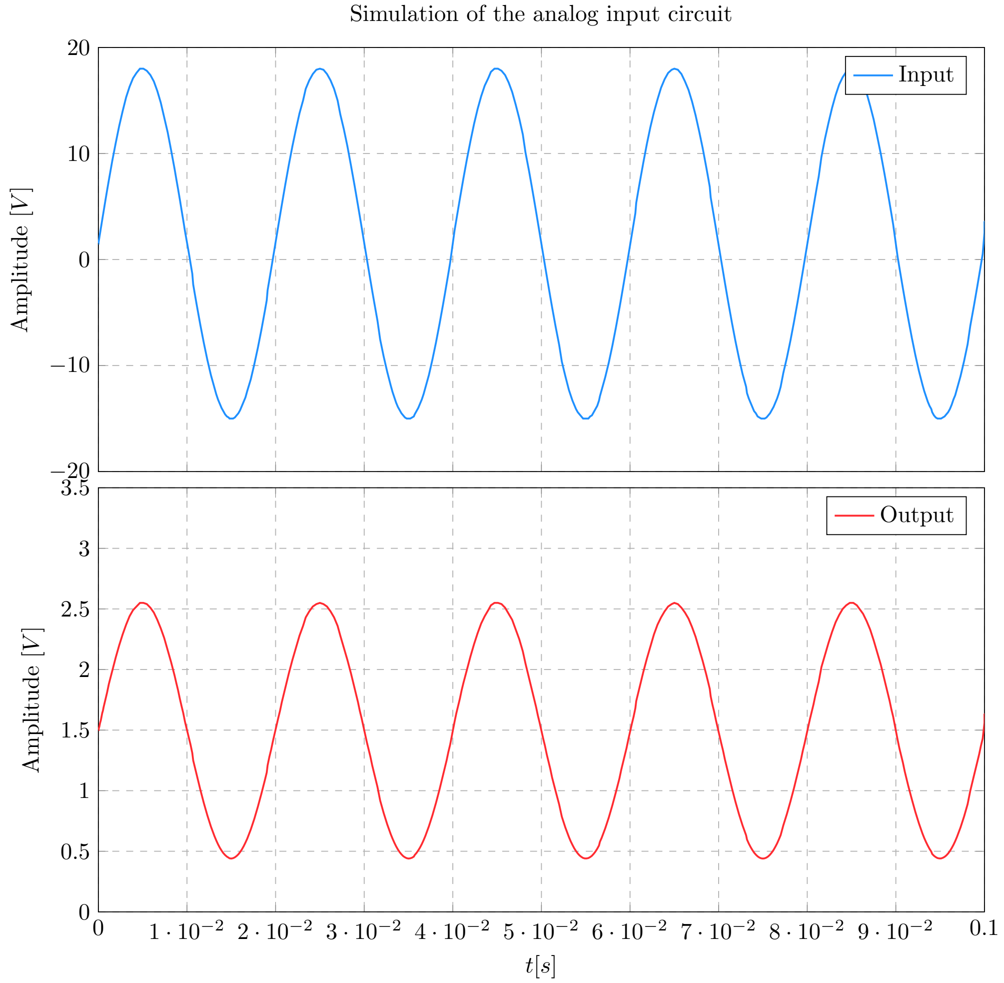


Figure 4.21: Simulation of the circuit used to compress the analog outputs from OPAL-RT in the 0-3V range, which is compatible with the control card.

4.4.4 PCB debugging and signals compatibility tests

After the PCB was manufactured we tested the circuit to verify if it behaves as the LTspice simulations. To simulate the Analog Outputs from OPAL-RT we feed a sine wave with an amplitude of 16V to the Analog Inputs of the interface board and using to the test points we checked with an oscilloscope if the operational amplifier outputs were correct. We did the same to reproduce the Digital Outputs from OPAL-RT, feeding in this case a square wave with an amplitude of 12V. To test the Analog and Digital Inputs to OAPL-RT instead we connected the control card that produced a PWM modulation of a sine wave. Using the DIP switches that allow to route the PWM signals to the DACs or directly to the Digital Inputs of OPAL-RT, we tested the signal compatibility. All the tests gave a positive result, confirming that the interface board works properly. At this point it is possible to mount the Interface board in a rack with OPAL-RT, and connect the Ethernet cables to route the signals between the control card and the real-time simulator.

In Figure 4.22 the final setup used to perform CHIL simulations at the System Level is shown. At the bottom there is OPAL-RT connected to two interface boards via Ethernet cables.

Figure 4.22: Setup composed of OPAL-RT and the multiple interface boards, where each of them hosts a control card.

4.5 Conclusions

We started this chapter introducing by the idea of system level validation proceeding with a discussion that explained the importance of performing CHIL simulations at a system level. Since a setup able to perform this kind of real-time simulations has not yet been developed and configured, we focused our attention in the realization of such a system. Based on that we introduced a new high performance real-time simulator (OPAL-RT) specifically developed for performing demanding real-time simulations, describing its architecture and giving a detailed overview of its possibilities. Moreover, we explained how to configure OPAL-RT and the procedure to follow to build the models of the controllers and the network to be simulated. The chapter proceeds introducing different configurations suitable to perform system level validation of the control strategies presented in Chapter 2. In particular we focus our attention to the case in which the controllers run in the CPU, while the circuit model in the FPGA of the real-time simulator. Subsequently based on that we perform a simulation of a modified version of the nine-bus system, where all the converters are controlled using grid forming strategies. We proceeded considering a second scenario in witch the grid forming strategies are implemented inside external control boards while OPAL-RT simulates the behaviour of the network model. Such a configuration however cannot be realized without the development of a specific interface board that acts like a bridge between the real-time simulator and the control card. The interface card solves the principal issues related to the signals incompatibility between OPAL-RT and the control card. The chapter ends with the realization of the interface board, which is the last piece of the puzzle needed to complete the setup that is able to perform CHIL simulations allowing system level validation.

Chapter 5

Conclusions and future work

5.1 Conclusions

The work presented in this thesis provides a guideline for the configuration of two setups used to run CHIL simulations and a methodology to perform component and system level validations.

The thesis starts with a general overview of the European power energy system. The European politics is encouraging the development and adoption of renewable energy sources in order to reduce air pollution and climate change caused by the massive use of fossil fuels. This is leading to a remarkable increment of the wind and solar energy production. As a consequence, the electric grid is undergrounding a drastic change, i.e., a transition from a centralized network to a distributed network system. A distributed network system based on renewable energies, in particular solar and wind energy, presents a low-inertia which introduce new challenges in the control strategies in order to guarantee the grid stability. To address the transition towards a 100% renewable network novel grid forming techniques have been developed and are under test.

In Chapter 2 we presented four grid forming strategies, explaining their similarities with the traditional synchronous generators. From the study it becomes clear that how the rotational inertia of the synchronous generators guarantees grid stability. Therefore, the loss of the synchronous machines due to the increment of wind and PV farms affects the grid stability, because it reduces the rotational inertia of the grid. The grid forming strategies address the problem by emulating some aspects of the synchronous generators. In particular, we highlighted the similarities between the primary frequency droop control of a synchronous machine and the grid forming controllers. Furthermore, we provided a basic circuit model composed by an inverter, a simple dc and ac network which is used in the following chapters to perform CHIL simulations.

Once we presented the grid forming techniques from a theoretical point of view, we started developing and configuring a suitable setup to perform component level validation. In the literature it is possible to find several studies about grid forming techniques, but almost no implementations of them in real hardware. Therefore, we focused our attention on the configuration of a setup composed of a real-time simulator, a control card, and an interface board. At the beginning we presented the methodology used to prepare and simulate the circuit and controllers models. The circuit model is automatically compiled and loaded into the real-time simulator, while the controller model must be translated into c-code and loaded inside the control card manually. In order to facilitate the conversion of the controllers model, we developed a custom library in c-code, which implements the common blocks used in the PLECS software. Furthermore, we addressed all the problems related to signals compatibility between the different hardware parts and we proposed a possible solution to collect the data of the real-time simulations. A Python script was developed to automatically run the real-time simulations and to automatically collect the data. In this way we reduced as much as possible the steps required from the user to perform component level validation. The chapter ended presenting some results obtained running CHIL simulations of the four grid forming techniques described in Chapter 2. We highlighted the differences between the real-time simulations and the offline simulations, in particular referring to how the signals transmission delays affect the controllers behaviour. We also noticed that the data collected from the real-time simulations is affected by more noise in comparison with the offline simulations.

Once we successfully implemented and validated the grid forming techniques in the control card performing CHIL simulations, a naturally way to proceed is to integrate this "module" (control card + grid forming controllers) in a more complex grid. The interconnection of multiple grid forming converters also with synchronous generators allows to study the impact of renewable energies on the grid stability. Furthermore, it is possible to observe the behaviour of the grid forming converters if the grid losses a synchronous generator or if it drastically increases the power demand. Therefore, we moved our attention to system level validation.

One of the key points for us is to perform CHIL simulations of a large system maintaining the same accuracy level used for component level validation. To satisfy this requirement we introduced a more high performance real-time simulator, which can be connected to multiple control cards. First of all, we provided an overview of the capability of the real-time simulator, presenting different configurations useful to simulate synchronous generators, power electronics and grid forming converters. In the first scenario, we tested a modified version of the Nine-Bus system. The controllers and the power hardware were implemented inside the real-time simulator, taking advantage of its CPU+FPGA architecture that for a first approach allows to test the Nine-Bus system without the control cards. After successfully implementing the Nine-Bus system composed by three grid forming converters in the real-time simulator, we collected the data and compared it to the results obtained with the offline simulation. Secondly, we considered an additional scenario where the grid forming converters are implemented inside control cards, while the circuit model is implemented inside the real-time simulator. However, the control cards and the real-time simulator are not electrically compatible. For that reason we built a custom interface keeping in mind three important aspects: flexibility, modularity, and user-friendliness. With the interface board we also avoid electrical issues that can be caused by a not correct setup of the signal voltage levels on the real-time simulator or on the control board side. The signal voltage level compatibility is guaranteed via hardware. The thesis work ends with the realization of the interface board and the initial tests for properer functionality.

Overall, with this thesis:

- we successfully configured a setup to preform component level validation.
- we provided a methodology to help the user run CHIL simulations.
- we developed PLECS and c-code libraries to help the user translate the controllers model to c-code.
- we developed a Python script to automatically run the CHIL simulations and collect the data.
- we successfully configured a setup to preform system level validation.
- we developed an interface board to connect multiple control cards to the real-time simulator.

5.2 Future work

In this thesis we prepared and properly configured two different platform used to perform component and system level validation. After implementing a simple model and verifying that both platforms work properly, we left for future work the simulation of more complex scenarios.

With the setup used to perform component level validation it is interesting to further investigate the behaviour of the grid forming controllers when connected to an infinite bus, to a constant current/power load, and to a synchronous machine. Furthermore, it is interesting to connect the converter to an unbalanced three phase grid in terms of voltage and phase. It is also possible to work on the dc side, modelling differently the dc network to better represent PV or wind power sources.

Regarding the control card we suggest to use both the cores of the chip. The first core can be used to update the controllers, and manege the IO signals, while the second one can be used for communication purposes. Since we store the collected data in a RAM block that is shared between the two cores, the second core has access to it and can send the data via serial port to a host PC. In this way we avoid overloading the first core.

With the setup used to perform system level validation it is interesting to substitute a grid forming converter with a synchronous machine that could be emulated in the CPU of the real-time simulator and perform CHIL simulations of the test-cases presented in [8]. Furthermore, performing CHIL simulations of multiple control cards connected to the real-time simulator is something that can reveal interesting aspects regarding the interaction of digital controllers that actuate on the same system. The variety of test cases is very large and the user has the possibility to configure the platform to satisfy a broad spectrum of requirements.

Bibliography

[1] "AIT Austrian Institute of Technology GmbH." Online available https://www.ait.ac.at.

[2] "Renewable energy in Europe - 2018 - Recent growth and knock-on effects." https://www.eea.europa.eu/publications/renewable-energy-in-europe-2018.

[3] "Centralized Generation of Electricity and its Impacts on the Environment." https://www.epa.gov/energy/centralized-generation-electricity-and-its-impacts-environment.

[4] M. Rezkalla, M. Marinelli, M. Pertl, and K. Heussen, "Trade-off analysis of virtual inertia and fast primary frequency control during frequency transients in a converter dominated network," *IEEE PES Innovative Smart Grid Technologies Asia*, Dec. 2016.

[5] Q.-C. Zhong and G. Weiss, "Synchronverters: Inverters that mimic synchronous generators," *IEEE Transactions on Industrial Electronics*, vol. 58(4), pp. 1259 – 1267, May 2011.

[6] C. Arghir, T. Jouini, and F. Dörfler, "Grid-forming control for power converters based on matching of synchronous machines," *Automatica*, vol. 95, Apr. 2018.

[7] G.-S. Seo, M. Colombino, I. Subotic, B. Johnson, D. Groß, and F. Dörfler, "Dispatchable virtual oscillator control for decentralized inverter-dominated power systems: Analysis and experiments," in *2019 IEEE Applied Power Electronics Conference and Exposition (APEC)*, Mar. 2018.

[8] A. Tayyebi, D. Groß, A. Anta, F. Kupzog, and F. Dörfler, "Interactions of grid-forming power converters and synchronous machines - a comparative study," Feb. 2019.

[9] A. Crivellaro, A. Tayyebi, C. Gavriluta, D. Groß, A. Anta, F. Kupzog, and F. Dörfler, "Beyond low-inertia systems: Massive integration of grid-forming power converters in transmission grids," Nov. 2019.

[10] A. Tayyebi, D. Groß, A. Anta, F. Kupzog, and F. Dörfler, "Interactions of grid-forming power converters and synchronous machines â a comparative study," June 2019.

[11] G. Andersson, *Dynamics and Control of Electric Power Systems.* February 2012. Lecture 227-0528-00, ITET ETH.

[12] I. Khan, Y. Xu, and B. Tahir, "Design and manufacturing of digital mosfet based-avr for synchronous generator," in *2015 IEEE International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, June 2015.

[13] B. Johnson, M. Rodriguez, M. Sinha, and S. Dhople, "Comparison of virtual oscillator and droop control," July 2017.

[14] M. Sinha, F. Dörfler, B. B. Johnson, and S. V. Dhople, "Uncovering droop control laws embedded within the nonlinear dynamics of van der pol oscillators," *IEEE Transactions on Control of Network Systems*, vol. 4, pp. 347–358, Nov. 2014.

[15] H. Bevrani, T. Ise, and Y. Miura, "Virtual synchronous generators: A survey and new perspectives," *International Journal of Electrical Power Energy Systems*, vol. 54, pp. 244–254, Jan. 2014.

[16] S. D'Arco and J. A. Suul, "Virtual synchronous machines â classification of implementations and analysis of equivalence to droop controllers for microgrids," June 2013.

[17] L. Huang, H. Xin, Z. Wang, K. Wu, H. Wang, J. Hu, and C. Lu, "A virtual synchronous control for voltage source converters utilizing dynamics of dc-link capacitor to realize self-synchronization," Aug. 2017.

[18] L. Huang, H. Xin, Z. Wang, K. Wu, H. Wang, J. Hu, and C. Lu, "A virtual synchronous control for voltage-source converters utilizing dynamics of dc-link capacitor to realize self-synchronization," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 5, pp. 1565 – 1577, Aug. 2017.

[19] B. Johnson, S. Dhople, J. Cale, and A. Hamadeh, "Oscillator-based inverter control for islanded three-phase microgrids," *IEEE Journal of Photovoltaics*, vol. 4, pp. 387–395, Jan. 2014.

[20] M. Colombino, D. Groß, J.-S. Brouillon, and F. Dörfler, "Global phase and magnitude synchronization of coupled oscillators with application to the control of grid-forming power inverters," *IEEE Transactions on Automatic Control*, vol. 64, pp. 4496 – 4511, Oct. 2017.

[21] G.-S. Seo, M. Colombino, I. Subotic, B. Johnson, D. Groß, and F. Dörfler, "Dispatchable virtual oscillator control for decentralized inverter-dominated power systems: Analysis and experiments," *IEEE Applied Power Electronics Conference and Exposition (APEC)*, Nov. 2018.

[22] D. Groß, M. Colombino, J.-S. Brouillon, and F. Dörfler, "The effect of transmission-line dynamics on grid-forming dispatchable virtual oscillator control," Feb. 2018.

[23] "RT Box Manual." Online available https://www.plexim.com/sites/default/files/rtboxmanual.pdf.

[24] "TMS320F2837xD Delfino Microcontrollers - Technical Reference Manual." Online available https://www.mouser.com/datasheet/2/405/spruhm8a-340021.pdf.

[25] "RT Box Homepage." Online available https://www.plexim.com/products/rt_box.

[26] "PLECS Software." Online available https://www.plexim.com/products/plecs_standalone.

[27] A. Marinopoulos, F. Papandrea, M. Reza, S. Norrga, F. Spertino, and R. Napoli, "Grid integration aspects of large solar pv installations: Lvrt capability and reactive power/voltage support requirements," in *IEEE PowerTech 2011*, June 2011.

[28] "Node-RED." Online available https://nodered.org.

[29] "OPAL RT Hompage." Online available https://www.opal-rt.com.

[30] "OP5700 RCP/HIL/ FPGA-BASED REAL-TIME SIMULATOR USER MANUAL." https://blob.opal-rt.com/medias/L00161_0337.pdf.

[31] "ULN2803A Darlington Transistor Arrays." http://www.ti.com/lit/ds/symlink/uln2803a.pdf.

[32] "Inverting Buffer MC74AC541." https://www.onsemi.com/pub/Collateral/MC74AC540-D.PDF.

[33] "Digital to analog converter LTC2645." https://www.analog.com/media/en/technical-documentation/data-sheets/LTC2645.pdf.

[34] "Operational amplifier LTC6252." https://www.analog.com/media/en/technical-documentation/data-sheets/625234fc.pdf.

List of Tables

List of Figures