POLITECNICO DI TORINO

Master degree course in Mechatronic Engineering

Master Degree Thesis

Advanced Control of a Non Inverting Buck Boost Converter

Design, comparison and implementation of different control techniques for DC-DC converter

Supervisors
Prof. Marcello Chiaberge
Eng. Dario Gandini

Candidate
Arturo Rivelli
matricola: 277464

A.a 2020/2021

Contents

Chapter 1

Introduction

Nowadays, the energy conversion is a crucial point of investigation that interests various engineering fields of research. Solar energy, wind energy, thermal energy: generally, the main objective with the so called renewable energy is to efficiently convert it into electrical energy for general purpose. As a matter of facts, the electrical energy is extremely versatile and so, disposing of renewable sources of conversion is fundamental in order to reduce the pollution related to the combustion.

This study is linked to a particular type of "energy converters", the Thermoelectric generators (**TEGs**): as described in [1], TEGs are used in automotive applications for recovering waste heat of exhaust gas and convert it into electrical energy. This is quite useful in terms of reduction of fuel consumption, thus because the alternator work is reduced and, accordingly, the efficiency of the combustion engine is improved. However, the electrical energy produced has to be adapted to the load that will use it, that can be, for example, a battery: in particular, it needs a current converter to track the maximum power point of operation, in order to waste the minimum amount of the produced energy.

In this regard, DC-DC converters come to help: electrical circuits known as power converters, that can be used to transform a DC input voltage in a DC output voltage, higher or lower than the input, keeping the power constant. They are complex circuits, composed mainly by reactive components, such as capacitors and inductors, switches and diodes. They are generally divided into two main categories: linear regulators and switching-mode power supplies (**SMPS**). Linear regulators were widely used in the past, thanks to the low ripple introduced in the signals and the fast transient response obtained. However, SMPS are preferred: they can have a very high efficiency factor (between 75 and 95%) and they are available in a lot of different topologies, among which the most common are buck, boost and buck-boost converters.

In order to drive the input power of the TEG, a particular variation of the buck boost converter is exploited: the Non Inverting Buck Boost Converter (**NIBB**), as described in [2]. This topology allows to transfer energy with a very high efficiency

and to provide an output voltage that can be higher or lower than the input voltage supplied.

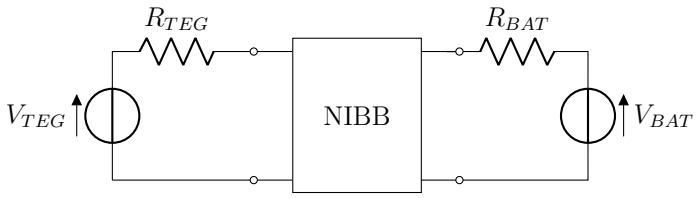


Figure 1.1: Schematic representation of the proposed model for the plant

The main objective of this study is to efficiently control the NIBB, analysing and comparing different control techniques by means of simulations, using Matlab and Simulink, and through testing on a real-time hardware. The initial simulations are provided representing the TEG and the battery as Thevenin equivalents: this approximation is important in order to simplify the control design procedure, focusing on the main object of the work.

1.1 Problem statement

To define a controller law for this type of converter can be a difficult task: the NIBB is a non-linear and time-variant system, with a transfer function representation that changes depending on the working condition in which the circuit is.

The exploited topology is characterized by four switches that are driven by two different command inputs (they are two couples of complementary switches), leading to a MISO system problem where the control variable is generally identified as the output voltage. This complexity can be handled by using a dual carrier system, that allows to define a unique signal able to drive the two command inputs separately. In this way, the system can be seen as a SISO system, without introducing any particular problem.

Moreover, it is important to outline the difficulty in using the output voltage as feedback control signal: this problem is related to the ratio between the input and the output impedance of the converter than can lead to instability, as clearly described by [3].

1.2 Proposed solutions

In the literature, there exist various control laws that can be successfully applied to DC-DC converters. The first techniques used for the control of these circuits were developed using analog comparators: taking into account the transfer function of

the converter, a compensator circuit is build, in order to shape the open loop transfer function of the system as a simple integrator. This allows to obtain a good tracking of the reference, with performances that depend mainly on the quality of the components and on the definition of the compensation itself.

Generally, all the controllers applied to the converter are divided into two categories: voltage mode and current mode controllers. The former is based on using the output voltage of the converter as control variable; the latter uses an inner loop that aims to control the peak (or the valley or both) of the inductor current. A voltage mode control is simple to be designed, but it requires to pay attention to the transfer function obtained, that can change with the input voltage provided. For the current mode, instead, because the reference for the current is derived from comparing the output voltage with the desired reference voltage, a double loop structure is provided. With the current mode, the controller is more robust thanks to the presence of both an external and an internal control loop. The current control loop, instead, allows to define a transfer function that does not depend on the input voltage, even if it could be problematic to be used, due to the phenomenon of sub-harmonic instability. A complete explanation about analogical control of DC converters is provided by [4] and [5].

Nowadays, thanks to the possibility of digital implementation, different and more complex control techniques can be easily simulated and then tested using microcontrollers ([6] provides an interesting comparison that outlines the multiples advantages from the usage of digital controllers, such as the possibility of simulate the system through Matlab/Simulink). In the literature, the main focus is to control either the output voltage or the inductor current: instead, in this work the input current provided by the generator is exploited as control variable. As mentioned above, the control of the output voltage is quite difficult in this topology, due to the instability effects related to the output impedance. Moreover, in the majority of works, a pure resistive component is considered as load: this is a huge difference with respect to the model used for this project, where the output is a Thevenin equivalent, that is a series of a resistor with a voltage generator.

1.3 Thesis objective

The objective of this thesis dissertation is to develop and compare different digital control techniques able to efficiently handle the power tracking of the NIBB converter, robustly and with acceptable performances. In particular, the main objectives are:

- to study the actual topology of the converter and define a suitable model for the true plant;
- to develop and compare through simulation different control techniques, both in continuous and discrete time;

- to test the developed control algorithms on a real hardware prototype.

1.4 Thesis structure

In order to accomplish the study objectives, this dissertation is structured as follows:

- chapter 2 provides a general description of the NIBB converter. Starting from that, a suitable model for the control design is derived, outlining both the state equations and the necessary transfer functions;
- in chapter 3, the design of a PI controller is outline. The PI is the starting point that allows to outline the possible advantages provided by a more complex control technique. The design is developed for both continuous and discrete time, paying attention to the simulation results and the step performances;
- from chapter 4 to chapter 6, classical control techniques are studied, exploiting H_∞ design, linear quadratic regulator and sliding mode control. Each controller is designed both in continuous time and discrete time, analyzing the performances of the system by means of Matlab simulations;
- in chapter 7 an innovative control technique is introduced, that is the fuzzy logic controller. This type of controller acts imitating the human behavior, without needing of a model to be known. It is directly implemented in discrete time, showing interesting results with respect to the classical control techniques;
- chapter 8, instead, introduces a filtering and estimation stage, exploiting the Kalman filter theory. A simple extended Kalman filter is designed and then a modified structure with two filtering stages is described;
- in chapter 9, all the designed controllers are deployed in hardware, paying attention not only to controller and filter performances but also to the execution time of the algorithms;
- finally, chapter 10 carries out a complete critical analysis about the obtained results, proposing possible improvements that may be developed in future.

In the appendix of this dissertation are presented a short description of the Kalman filter theory and the code of the `time_performance` function, a custom Matlab function exploited for the analysis of the controllers performances.

Chapter 2

Model and transfer functions

The first step for the control design is the definition of a model, a suitable mathematical representation that describes the relationships between the different components of the system. In this chapter, a brief description of the employed topology is outlined, focusing on the mathematical model of the system, that is represented as a state space set of equations and developed through small signals analysis. Then, in order to treat the multi-input system as a single-input system, the dual carrier modulation is introduced, reducing the number of command inputs to one and so simplifying the control design procedure. Finally, a very simple validation of the model is carried out, comparing a simulation of the real circuit with the mathematical equations provided.

2.1 Circuit topology

The NIBB converter is a versatile topology, that allows to perform step-up (boost) and step-down (buck) functions within a single stage.

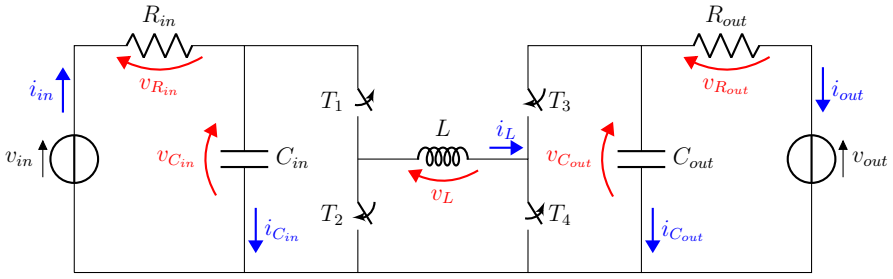


Figure 2.1: Non-Inverting Buck Boost converter schematic

This topology consists of two switching legs and one inductor in between: the first leg is driven by a PWM waveform with duty cycle d_a, while the second leg is driven by another duty cycle d_b, that can be different from or equal to d_a, depending on the type of modulation used. Generally, the switching of the first leg is related to the buck function (or buck mode), while the switching of the second leg is related to the boost function (or boost mode): if only the first leg is switching, the NIBB is working in buck mode, instead, if only the second leg is switching, the NIBB is said to work in boost mode. If both legs are switching, the system is working in buck-boost mode.

The way of functioning of the circuit is called "modulation". Depending on how the legs are driven, there exist three different types of modulation:

- **One-mode modulation**: there is only one duty cycle, identical for the two switching legs ($d_a = d_b$), that allows the system to work only in buck-boost mode;
- **Two-modes modulation**: there are two different duty cycles, one for the first leg and one for the second leg. This is a very efficient mode in terms of power losses, because the buck-boost area is avoided, with only one couple of working switches at a time;
- **Three-modes modulation**: it can be seen as a combination of the first two modulations, in which the transition between buck and boost is preceded by a buck-boost working area. This type of modulation has more switching losses, due to the presence of the buck-boost mode. However, it is very convenient, because it introduces a smooth transition between the buck and the boost area.

In this application, the three-modes modulation is applied, with a Continuous Current Mode (**CCM**) functioning (the circuits components are chosen in order to avoid the inductor current to cross the zero). A comprehensive description of the NIBB is provided by [7]. Moreover, it is important to outline that, in the buck-boost area, due to gate delays and dead-times, some duty cycles cannot be produced, creating dead-zones. Some strategies have been developed in order to avoid this problem, as described in [8].

2.2 Modeling and state space representation

The common practice for DC-DC converters modeling is to use an ideal voltage generator as input and a resistive load as output [9]. Sometimes also the components' parasitic resistances are taken into account [10], [11]. However, this ideality could be extremely distant from real application with renewable sources and recovery systems. For this reason, simple Thevenin equivalents are used for the input

voltage source and the output load, taking into account for the series resistors of both the generator and the battery (see figure 2.1).

The control of the system is provided using as command inputs the two duty cycles d_a and d_b. The buck phase is related to the functioning of the switches T_1 and T_2, with T_1 driven by d_a and T_2 by the complementary $\overline{d_a}$. The boost phase, instead, is related to the functioning of the switches T_3 and T_4, with T_4 driven by d_b and T_3 by the complementary $\overline{d_b}$. T_1 and T_2 form the so called "buck leg", while T_3 and T_4 define the "boost leg". For the modeling, dead zones effects are neglected.

In order to derive a suitable model for the circuit, it is important to pay attention to the nature of the system itself. As a matter of facts, a switch mode converter can be defined as a non-linear and time-variant system, thus because of the switches, that are indeed non-linear and time-variant components. The first step in the definition of a suitable NIBB model is to overcome the time-variance. This could be accomplished exploiting the state space averaging, a common procedure in the power electronics field, that consists in averaging the states of the system. Looking at the functioning of the converter, four different states can be defined, in relation with the different combinations of switches configurations. Kirchhoff's laws can be applied in order to derive four sets of equations:

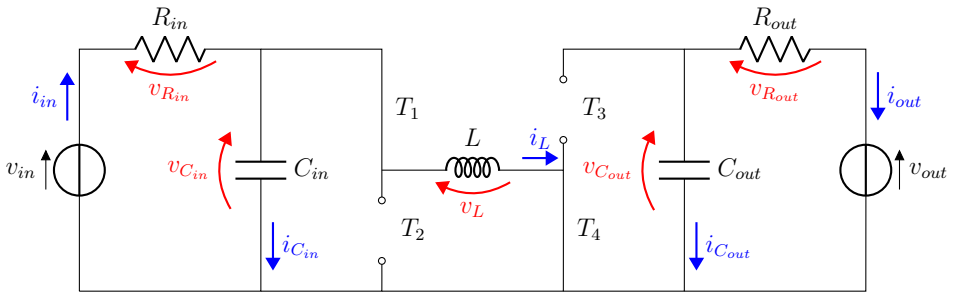


Figure 2.2: Configuration A, with $d_a = 1$ and $d_b = 1$

$$C_A : \begin{cases} \dot{v}_{C_{in}} = \frac{i_{in} - i_L}{C_{in}} \\ \dot{v}_{C_{out}} = \frac{-i_{out}}{C_{out}} \\ \dot{i}_L = \frac{v_{C_{in}}}{L} \end{cases} \tag{2.1}$$

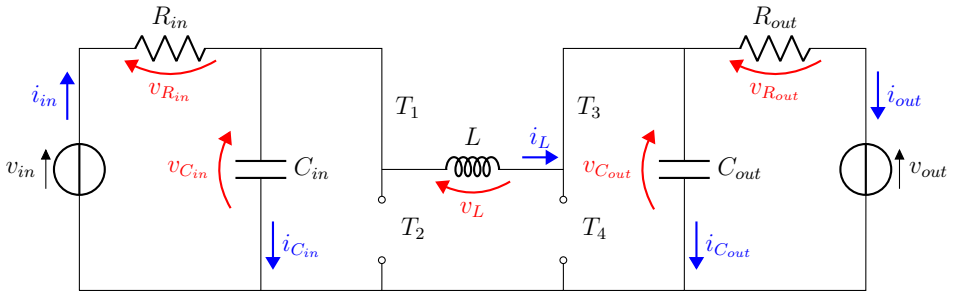


Figure 2.3: Configuration B, with $d_a = 1$ and $d_b = 0$

$$C_B : \begin{cases} \dot{v}_{C_{in}} = \frac{i_{in} - i_L}{C_{in}} \\ \dot{v}_{C_{out}} = \frac{i_L - i_{out}}{C_{out}} \\ \dot{i}_L = \frac{v_{C_{in}} - v_{C_{out}}}{L} \end{cases} \tag{2.2}$$

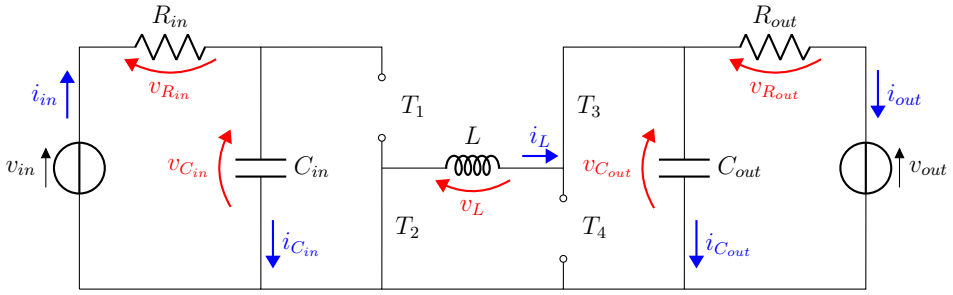


Figure 2.4: Configuration C, with $d_a = 0$ and $d_b = 0$

$$C_C : \begin{cases} \dot{v}_{C_{in}} = \frac{i_{in}}{C_{in}} \\ \dot{v}_{C_{out}} = \frac{i_L - i_{out}}{C_{out}} \\ \dot{i}_L = \frac{-v_{C_{out}}}{L} \end{cases} \tag{2.3}$$

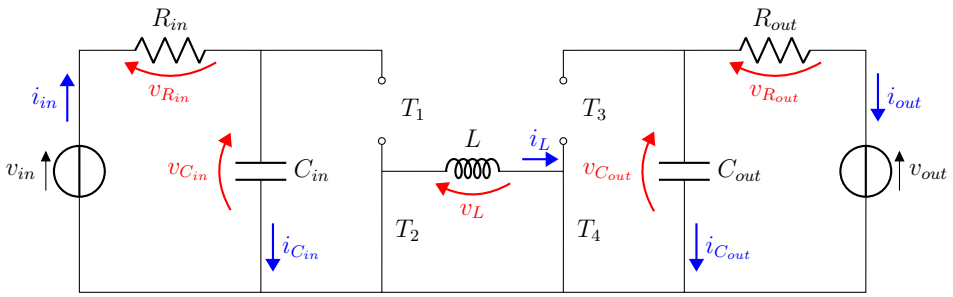


Figure 2.5: Configuration D, with $d_a = 0$ and $d_b = 1$

$$C_D : \begin{cases} \dot{v}_{C_{in}} = \frac{i_{in}}{C_{in}} \\ \dot{v}_{C_{out}} = \frac{-i_{out}}{C_{out}} \\ \dot{i}_L = 0 \end{cases} \tag{2.4}$$

C_A and C_B define a boost converter configuration; C_B and C_C define a buck converter configuration; C_D disconnects the inductor from both the input and the output stage of the circuit. The input and the output currents are linked to the state variables, as expressed in equations (2.5) and (2.6).

$$i_{in} = \frac{v_{in} - v_{C_{in}}}{R_{in}} \tag{2.5}$$

$$i_{out} = \frac{v_{C_{out}} - v_{out}}{R_{out}} \tag{2.6}$$

Therefore, the model can be written as a state space representation, where the output vector is $z = [i_{in} \;\; i_{out}]$, the input vector is $v = [v_{in} \;\; v_{out}]$ and the state vector is $x = [v_{C_{in}} \;\; v_{C_{out}} \;\; i_L]$.

$$\begin{cases} \dot{x} = A \cdot x + B \cdot v \\ z = C \cdot x + D \cdot v \end{cases} \tag{2.7}$$

The state matrices of (2.7) can be obtained by averaging the states as shown in equation (2.8):

$$\dot{x} = d_a d_b C_A + d_a \overline{d_b} C_B + \overline{d_a d_b} C_C + \overline{d_a} d_b C_D \tag{2.8}$$

Finally, using the four sets of equations (2.1) - (2.4) and substituting i_{in} and i_{out} with the relations (2.5) and (2.6), the matrices A,B,C and D are defined as:

$$A = \begin{bmatrix} -\frac{1}{R_{in}C_{in}} & 0 & -\frac{d_a}{C_{in}} \\ 0 & -\frac{1}{R_{out}C_{out}} & \frac{\overline{d_b}}{C_{out}} \\ \frac{d_a}{L} & -\frac{\overline{d_b}}{L} & 0 \end{bmatrix} \quad B = \begin{bmatrix} \frac{1}{R_{in}C_{in}} & 0 \\ 0 & \frac{1}{R_{out}C_{out}} \\ 0 & 0 \end{bmatrix} \tag{2.9}$$

$$C = \begin{bmatrix} -\frac{1}{R_{in}} & 0 & 0 \\ 0 & \frac{1}{R_{out}} & 0 \end{bmatrix} \quad D = \begin{bmatrix} \frac{1}{R_{in}} & 0 \\ 0 & \frac{1}{R_{out}} \end{bmatrix} \tag{2.10}$$

In this state representation, v is a set of exogenous inputs that can be treated as disturbances, while the command input, composed by d_a and d_b, appears inside the matrix A, making the obtained representation non-linear. Another important remark is related to the choice of the control output: indeed, there are two current outputs that can be exploited as control variable, that are i_{out} and i_{in}.

2.3 Small-signals analysis and transfer function

In order to find a suitable transfer function of the plant, the small-signals analysis is exploited [4]. This technique allows to write the transfer function of a DC-DC converter by representing a general signal g as a composition of a "small signal" $\hat{g}$ and a "big signal" G.

$$g = G + \hat{g} \quad \text{with} \quad G \gg \hat{g}$$

The small signal represents the variable component of the signal itself, while the big signal can be seen as the constant component. Considering the matrices A and B shown in (2.9), the states are expressed as follows:

$$\frac{d}{dt}(V_{C_{in}} + \hat{v}_{C_{in}}) = -\frac{V_{C_{in}} + \hat{v}_{C_{in}}}{R_{in}C_{in}} - \frac{(D_a + \hat{d}_a)(I_L + \hat{i}_L)}{C_{in}} + \frac{V_{in} + \hat{v}_{in}}{R_{in}C_{in}} \tag{2.11}$$

$$\frac{d}{dt}(V_{C_{out}} + \hat{v}_{C_{out}}) = -\frac{V_{C_{out}} + \hat{v}_{C_{out}}}{R_{out}C_{out}} - \frac{(1 - D_b - \hat{d}_b)(I_L + \hat{i}_L)}{C_{out}} + \frac{V_{out} + \hat{v}_{out}}{R_{out}C_{out}} \tag{2.12}$$

$$\frac{d}{dt}(I_L + \hat{i}_L) = \frac{(V_{C_{in}} + \hat{v}_{C_{in}})(D_a + \hat{d}_a)}{L} - \frac{(V_{C_{out}} + \hat{v}_{C_{out}})(1 - D_b - \hat{d}_b)}{L} \tag{2.13}$$

The equations (2.11) - (2.13) can be analyzed from two points of view, considering either the big signals or the small signals alone. The big signals analysis

allows to obtain a set of relationships that expresses the steady-state behavior of the system.

$$V_{C_{out}} = V_{C_{in}} \cdot \frac{D_a}{1 - D_b} \tag{2.14}$$

$$I_{in} = D_a I_L \tag{2.15}$$

$$I_{out} = (1 - D_b) \cdot I_L \tag{2.16}$$

Using (2.15) and (2.16), it is possible to obtain a relation between I_{in} and I_{out}, as written in (2.17).

$$I_{in} = I_{out} \cdot \frac{D_a}{1 - D_b} \tag{2.17}$$

Instead, with the small signals analysis, it is possible to derive the transfer function of the system. The small-signals analysis consists in taking into account only the small signal components of equations (2.11) - (2.13), neglecting all the products between small signals themselves: this type of procedure can be seen as a linearization of the initial system of equations. If i_{out} is chosen as control variable, the transfer function of $v_{C_{out}}$ is needed.

$$\hat{v}_{C_{out}}(s) = R_{out} \cdot \frac{(b_1 s + b_0) \cdot \hat{d}_a + (c_2 s^2 + c_1 s + c_0) \cdot \hat{d}_b}{a_3 s^3 + a_2 s^2 + a_1 s + a_0} \tag{2.18}$$

All the transfer function coefficients are listed in table 2.1.

Coefficient	Value
a3	$C_{in}C_{out}LR_{in}R_{out}$
a2	$L(C_{in}R_{in} + C_{out}R_{out})$
a1	$L + R_{in}R_{out}(\overline{D_b}^2 C_{in} + D_a^2 C_{out})$
a0	$\overline{D_b}^2 R_{out} + D_a^2 R_{in}$
b1	$\overline{D_b} C_{in} R_{in} V_{C_{in}}$
b0	$\overline{D_b}(V_{C_{in}} - I_L R_{in} D_a)$
c2	$-C_{in} I_L L R_{in}$
c1	$\overline{D_b} C_{in} R_{in} V_{C_{out}} - I_L L$
c0	$\overline{D_b} V_{C_{out}} - D_a^2 I_L R_{in}$

Table 2.1: Coefficients of $\hat{v}_{C_{out}}(s)$ transfer function

The transfer function of $\hat{v}_{C_{out}}$ is computed by applying the super-position principle and by considering only the command inputs $\hat{d}_a$ and $\hat{d}_b$. The parameters of the transfer function change depending on the values of the big signals, that are defined by the "working condition" (also called "bias point") of the DC-DC converter. Moreover, the transfer function can be simplified if the converter is outside of the buck-boost area: as a matter of facts, in pure buck d_b has no variations, with $\hat{d}_b = 0$, while in pure boost d_a has no variations, with $\hat{d}_a = 0$.

Then, equations (2.18) and (2.6) can be used for deriving the transfer function of the output current:

$$\hat{i}_{out}(s) = \frac{\hat{v}_{C_{out}}(s)}{R_{out}}$$

So, by studying the transfer function related to the voltage on the output capacitor it is possible to obtain information about the output current and its usage for control. Focusing on the DC gain of (2.18) and using equations (2.15) and (2.16), it is possible to obtain:

- **Buck**: $\hat{i}_{out}^{DC} = \frac{\overline{D_b}(V_{C_{in}} - I_{in}R_{in})}{\overline{D_b}^2 R_{out} + D_a^2 R_{in}}$
- **Boost**: $\hat{i}_{out}^{DC} = \frac{D_a(V_{C_{in}} - I_{in}R_{in})}{\overline{D_b}^2 R_{out} + D_a^2 R_{in}}$

The sign of the transfer function is decided by the quantity $(V_{C_{in}} - I_{in}R_{in})$. Until $V_{C_{in}}$ is greater than $I_{in}R_{in}$, the DC gain is positive and the system is stable. When $(V_{C_{in}} - I_{in}R_{in})$ is equal to zero, the voltage drop on R_{in} is exactly equal to $V_{C_{in}}$, that is the maximum power point for the input generator. However, if the current is too big and the sign of the DC gain becomes negative, the feedback control system becomes unstable. This is the main reason because the usage of the output current i_{out} may be problematic in the control of the converter. As a matter of facts, the objective is to control efficiently the converter around the maximum power point and this cannot be easily obtained using i_{out} as control variable, thus because any oscillation around the maximum power point could lead to instability.

There exists the same problem if the inductor current i_L is chosen as controlled variable. Indeed, the transfer function between i_L and d_a, d_b is computed as:

$$\hat{i}_L(s) = \frac{(b_2 s^2 + b_1 s + b_0) \cdot \hat{d}_a + (c_2 s^2 + c_1 s + c_0) \cdot \hat{d}_b}{a_3 s^3 + a_2 s^2 + a_1 s + a_0} \tag{2.19}$$

The sign of the transfer function is chosen by the difference $(V_{C_{in}} - I_{in}R_{in})$ and so there is the same problem of instability previously introduced for $\hat{i}_{out}$.

Coefficient	Value
a3	$C_{in}C_{out}LR_{in}R_{out}$
a2	$L(C_{in}R_{in} + C_{out}R_{out})$
a1	$L + R_{in}R_{out}(\overline{D_b}^2 C_{in} + D_a^2 C_{out})$
a0	$\overline{D_b}^2 R_{out} + D_a^2 R_{in}$
b2	$C_{in}C_{out}LR_{in}R_{out}V_{C_{in}}$
b1	$V_{C_{in}}(C_{out}R_{out} + C_{in}R_{in}) - C_{out}D_a I_L R_{in}R_{out}$
b0	$V_{C_{in}} - I_L R_{in} D_a$
c2	$C_{in}C_{out}LR_{in}R_{out}V_{C_{out}}$
c1	$C_{in}R_{in}R_{out}I_L\overline{D_b} + V_{C_{out}}(C_{in}R_{in} + C_{out}R_{out})$
c0	$V_{C_{out}} - I_L R_{out}\overline{D_b}$

Table 2.2: Coefficients of $\hat{i}_L(s)$ transfer function

The discussion is different for $\hat{i}_{in}$. Its transfer function with respect to the command input is obtained through the transfer function of $\hat{v}_{C_{in}}$, that is:

$$\hat{v}_{C_{in}}(s) = R_{in} \cdot \frac{(b_2 s^2 + b_1 s + b_0) \cdot \hat{d}_a + (c_1 s + c_0) \cdot \hat{d}_b}{a_3 s^3 + a_2 s^2 + a_1 s + a_0} \tag{2.20}$$

Coefficient	Value
a3	$C_{in}C_{out}LR_{in}R_{out}$
a2	$L(C_{in}R_{in} + C_{out}R_{out})$
a1	$L + R_{in}R_{out}(\overline{D_b}^2 C_{in} + D_a^2 C_{out})$
a0	$\overline{D_b}^2 R_{out} + D_a^2 R_{in}$
b2	$-C_{out}I_L LR_{out}$
b1	$I_L L - D_a C_{out}R_{out}V_{C_{in}}$
b0	$-D_a V_{C_{in}} - \overline{D_b}^2 I_L R_{out}$
c1	$-D_a C_{out}R_{out}V_{C_{out}}$
c0	$D_a(-V_{C_{out}} - I_L R_{out}\overline{D_b})$

Table 2.3: Coefficients of $\hat{v}_{C_{in}}(s)$ transfer function

With (2.5), the transfer function of i_{in} is consequentially computed as:

$$\hat{i}_{in}(s) = -\frac{\hat{v}_{C_{in}}(s)}{R_{in}}$$

Analyzing the DC gain as previously done with $\hat{i}_{out}$ and $\hat{i}_L$ is possible to outline an important difference: the sign of the DC gain does not change.

- **Buck**: $\hat{i}_{in}^{DC} = \frac{\overline{D_b}(V_{C_{out}} + I_{out}R_{out})}{\overline{D_b}^2 R_{out} + D_a^2 R_{in}}$
- **Boost**: $\hat{i}_{in}^{DC} = \frac{D_a(V_{C_{out}} + I_{out}R_{out})}{\overline{D_b}^2 R_{out} + D_a^2 R_{in}}$

The sign of the DC gain depends on sums of positive quantities, both in buck and boost mode ($V_{C_{out}} + I_{out}R_{out}$) and so it will not change with the working point. This ensures to avoid the instability problem that affects i_{out} and i_L, making i_{in} a good candidate as control variable.

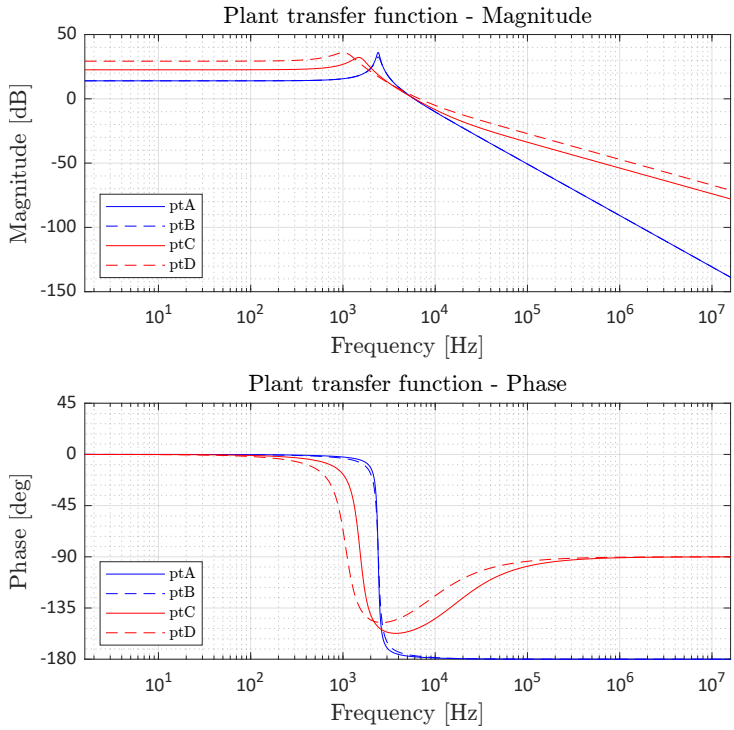


Figure 2.6: Transfer functions of $\hat{i}_{in}$ evaluated in four different working points, neglecting the effect of the ESRs

Figure 2.6 shows the transfer function of $\hat{i}_{in}$ evaluated in four different working points, called ptA, ptB, ptC, ptD, without taking into account the effect of the equivalent series resistors (**ESRs**) of the reactive components. The values associated with each point are resumed in table 2.4.

W.P.	D_a	D_b	V_{in} [V]	V_{out} [V]
A	1	0.6421	8	12
B	1	0.3831	15	12
C	0.6204	0	40	12
D	0.4314	0	60	12

Table 2.4: Testing working points, with duty cycles and voltage values

These points are chosen considering that the operative range of the converter is $v_{in} \in [8\ 60]$ V. ptA and ptB are boost working points, with transfer functions defined as $\frac{\hat{i}_{in}}{\hat{d}_b}$, while ptC and ptD are buck working points, with transfer function defined as $\frac{\hat{i}_{in}}{\hat{d}_a}$. In all the points, the condition of maximum power transmission has been applied (so the voltage on the input capacitor is always equal to half of the input voltage at the steady state, $V_{C_{in}} = \frac{V_{in}}{2}$).

If the ESRs are considered into the model, the mathematical representation becomes more complicated. Applying the same techniques previously exploited, the transfer functions of the same four testing points, indicated in table 2.4, can be constructed as shown in figure 2.7.

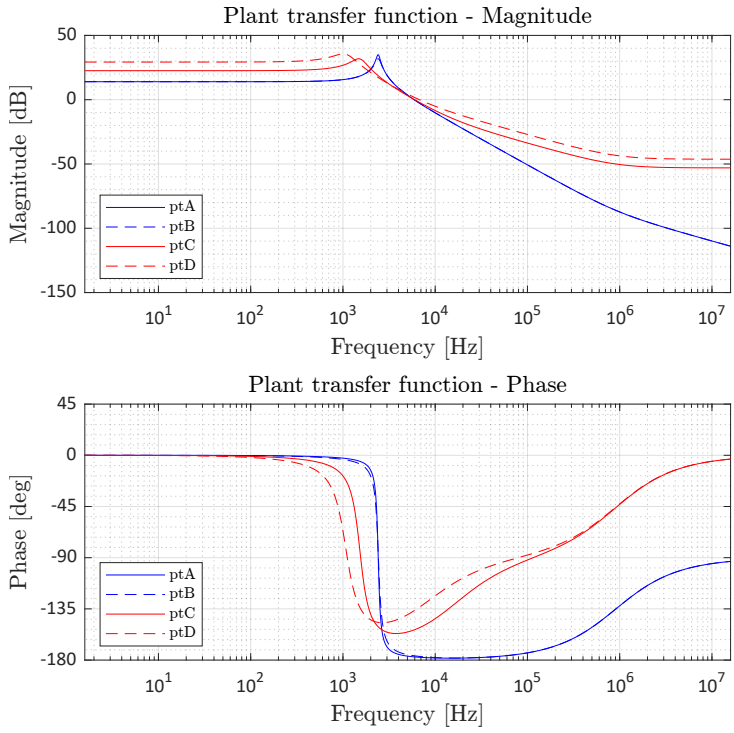


Figure 2.7: Transfer functions of $\hat{i}_{in}$ evaluated in four different working points, considering the effect of the ESRs

In both the cases, the different behavior between the buck and the boost area is outlined. ptA and ptB are very close to each other, overlapping in both the figure representations 2.6 and 2.7. Moreover, the transfer functions computed neglecting the ESRs effect are close to the same transfer functions computed with the ESRs effects, in the low/medium frequency range. The main difference is at high frequency, where an additional zero is added to the transfer function due to the ESR of the input capacitor.

$$\hat{i}_{in}(s) = -\frac{\hat{v}_{C_{in}}(s)}{R_{in}} \cdot (1 + sC_{in}ESR_{C_{in}})$$

The model that takes into account the ESRs is more precise. However, if necessary, the model without ESRs can be used as a good enough approximation. Indeed, for some applications, such as the LQR, the symbolical state space representation is needed: if the ESRs are included, the model is difficult to be handle (i.e. the state equation of the inductor current is significantly large) and so the design procedure becomes hard. For these cases, using the model without the ESRs, the representation of the plant is less accurate but simpler.

2.4 Dual-carrier modulation

The dual carrier modulation (analysed in detail in [12]) is an effective technique that can be used in order to handle both the command inputs of the two switching legs with just one unique signal. Even if this makes the total structure more complex, it allows to treat the plant as an **SISO** system, reducing the difficulty of an **MISO** system control design. Figure 2.8 can be used to explain clearly its working principle.

The comparison of the command signal u with two triangular signals W_1 and W_2 (i.e. the carriers) allows to obtain the command inputs of the two switching legs, identified as the PWM duty cycles d_a and d_b. The two carriers are bounded into two different sets, such that $W_1 \in [V_{1H}\ V_{1L}]$ and $W_2 \in [V_{2H}\ V_{2L}]$. Setting coherently the bounds of the carriers, it is possible to obtain an overlap region, where the command makes the system work in buck-boost mode (as indicated in figure 2.8). If the command input u is above V_{1H}, the system is working in pure boost, while if the command input u is below V_{2L}, the system is working in pure buck.

This approach has to be translated in a suitable mathematical representation, in order to be implemented in a digital controller. This can be reached imposing

the intersection between u and one carrier, for example W_1, at a certain time T_1:

$$\begin{aligned} u &= W_1(T_1) \\ &= V_{1L} + \frac{V_{1H} - V_{1L}}{T/2} \cdot T_1 \end{aligned} \tag{2.21}$$

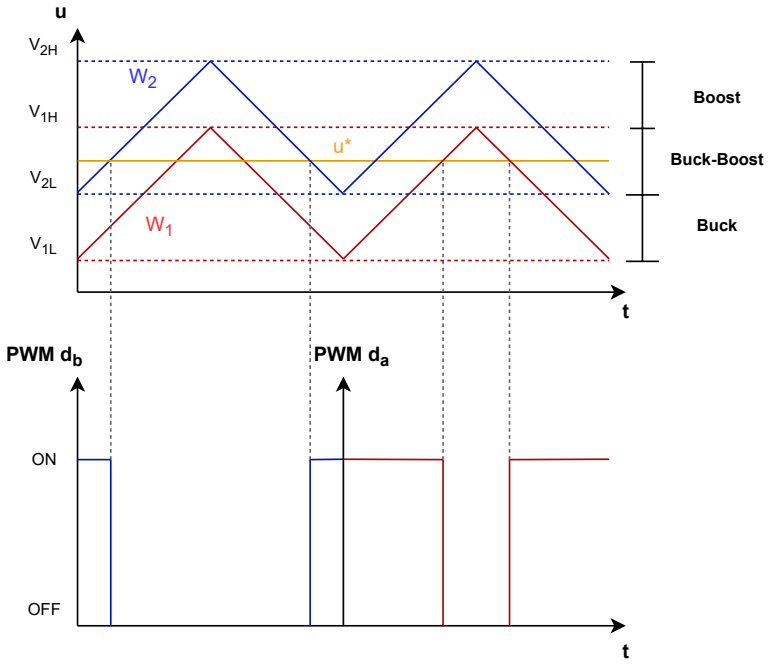


Figure 2.8: Graphical representation of the dual-carrier working principle

Because the triangle wave is symmetric, the duty cycle d_a is obtained as written in (2.22), where T_{ON} is the time during which the switch is ON, while T is the carrier period.

$$d_a = \frac{T_{ON}}{T} = \frac{2T_1}{T} \tag{2.22}$$

Combining equations (2.21) and (2.22), a relationship between d_a and u is then derived:

$$d_a = sat(u \cdot K1 + K2) \tag{2.23}$$

$$K1 = \frac{1}{V_{1H} - V_{1L}} \qquad K2 = \frac{-V_{1L}}{V_{1H} - V_{1L}}$$

The same reasoning is exploited for the duty cycle d_b, obtaining:

$$d_b = sat(u \cdot K3 + K4) \tag{2.24}$$

$$K3 = \frac{1}{V_{2H} - V_{2L}} \qquad K2 = \frac{-V_{2L}}{V_{2H} - V_{2L}}$$

The saturation function, in equations (2.23) and (2.24), is introduced in order to limit the value of the duty cycles between 0 and 1:

$$sat(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x \in [0,1] \\ 1 & \text{if } x > 0 \end{cases}$$

These relations are then implemented in a Simulink model, defining a digital PWM. As it is, this set of equations does not take into account for the insertion of dead-times, that are fundamentals in the real application in order to avoid dangerous short circuits in a switching leg. The dead-time is created adding a small value to the command input when compared with W_1 and W_2 in the creation of the duty cycles of T_2 and T_3 switches.This makes possible to obtain a smaller duty cycle for T_2 and T_3 that accounts for the dead-time zones.

An important remark has to be done in relation to the phase of the carriers. The phase shift between d_a and d_b affects the shape of the inductor current i_L in the buck-boost region of the converter, as also reported in [10] and [13]. The two legs can be synchronized (i.e. d_a and d_b are symmetric with respect to the same axis) or they can be shifted by an angle ϕ, with a maximum amount of shift of 180 deg. The different shape of i_L translates in different peak-to-peak inductor current $i_{L_{pp}}$, that is associated with some power losses in the converter. The difference between the two extreme cases ($\phi = 0$ deg and $\phi = 180$ deg) is well represented in figure 2.9, where the buck-boost buck mode and the buck-boost boost mode are compared.

The resulting $i_{L_{pp}}$ is computed as:

$$i_{L_{pp}}^{sync} = max\left(\frac{V_{out}}{L} \cdot \overline{d_a}, \frac{V_{in}}{L} \cdot d_b\right)$$

$$i_{L_{pp}}^{shift} = max\left(\frac{V_{out}}{L} \cdot (\overline{d_a} - d_b), \frac{V_{in}}{L} \cdot (d_b - \overline{d_a})\right)$$

The reduction of the peak is related to the configuration C_D, in which the inductor is short-circuited. This configuration is not produced when the two carriers are synchronized and it leads to a significant reduction of the ripple, leaving the average inductor current unchanged. This means that the shift does not affect the validity of the model, allowing to reduce the peak current on the inductor and, consequentially, the power losses related to the buck boost working area.

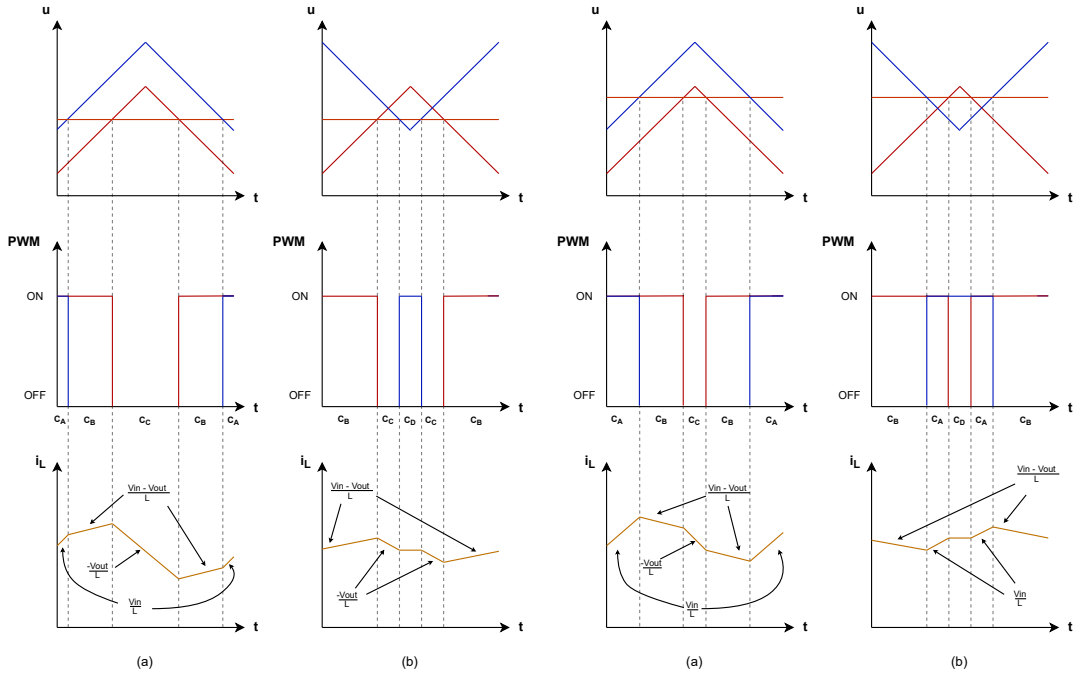


Figure 2.9: Inductor current analysis in buck-boost mode for (a) synchronous and (b) shifted case. On the left the buck-boost buck mode is represented, while, on the right, the buck-boost boost mode is shown

2.5 Model validation

In order to see how much accurate is the mathematical model of the converter obtained without considering the ESRs effect, a simple model validation is carried out. This operation is performed exploiting a simulation on Simulink of the real circuit of interest and comparing it with the mathematical model derived from the state space equations. As a matter of facts, there is a small difference if we consider the real circuit itself, however a simulated circuit is able to provide relevant information if it is enough precise.

The simulation of the NIBB converter is provided in Simulink using the Simscape environment. As represented in figure 2.10, which shows the Simulink diagram used, the circuit is build as much complete as possible, with the ESRs and the transistors instead of the ideal switches (each transistor is characterised by an internal resistance that can be set in the Simulink model).

The values of all the components are listed in table 2.5. The transistor used for the switches are MOSFET, in which the FET resistance $R_{ds_{ON}}$ and the Diode resistance R_d values can be specified, along with the diode forward voltage V_{df}. The switching frequency is set as equal to $f_{SW} = 150\,\text{kHz}$.

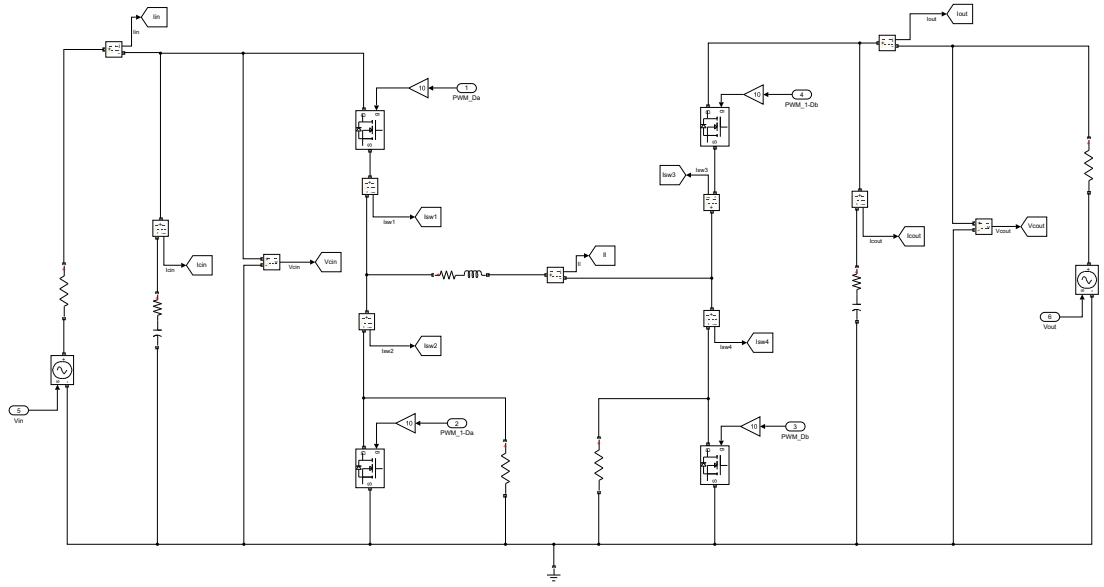


Figure 2.10: Simulink Model of the NIBB converter

Component	Value	Unit
R_{in}	2.4	Ω
C_{in}	437	μF
$ESR_{C_{in}}$	0.4	$m\Omega$
R_{out}	22.5	$m\Omega$
C_{out}	437	μF
$ESR_{C_{in}}$	0.4	$m\Omega$
L	10	μH
ESR_L	1.11	$m\Omega$
$R_{ds_{ON}}$	3	$m\Omega$
V_{df}	1	V

Table 2.5: Parameters' values of the Simulink model in figure 2.10

Then, also the state space model without ESRs is build in Simulink, exploiting the Matlab function blocks in order to implement the set of state and output equations ((2.9) and (2.10)) and the PWM working principle ((2.23) and (2.24)). The values of the components specified in the functions are exactly the same used for the simulated converter.

In order to describe the goodness of the model, four evaluation indices are used: the Mean Absolute Error (**MAE**), the Root Mean Squared Error (**RMSE**), the Relative Absolute Error (**RAE**) and the Relative Squared Error (**RSE**).

$$MAE = \frac{1}{N} \cdot \sum_{i=1}^{N} |e_i| \qquad RMSE = \sqrt{\frac{\sum_{i=1}^{N} (e_i)^2}{N}}$$

$$RAE = \frac{\sum_{i=1}^{N} |e_i|}{\sum_{i=1}^{N} |y_i - \overline{y}|} \qquad RSE = \frac{\sum_{i=1}^{N} (e_i)^2}{\sum_{i=1}^{N} (y_i - \overline{y})^2}$$

e_i is the model error, computed as $e_i = y_i - \hat{y}_i$, where y_i is the measure provided by the Simulink model and $\hat{y}_i$ is the corresponding one obtained using the mathematical model. Instead, $\overline{y}$ is the mean value of the whole set of measures y_i.

MAE and RMSE can be compared to the concept of "absolute error": they measure the distance between the measure and the model data. They are scale dependent and so they cannot be used to compare quantities with different scales. On the other hand, the RAE and the RSE can be seen as "relative errors" that are independent from the scale and that can be used for comparison between different quantities, with different scales.

In order to have a comprehensive evaluation of the goodness of the model, four working points are considered. The results of the analysis are reported in tables 2.6 - 2.9. Also, a visual comparison related to i_{in} and i_{out} is provided in figure 2.11 - 2.14. All the results are obtained considering a simulation time of 0.02 s, with a sampling frequency of 1000 kHz and a total number of 20000 samples.

The information provided by the indices are significant: the RMSE and the MAE values increase going from boost to buck working mode. This is mainly related to the fact that RMSE and MAE are scale-dependent. Indeed, from boost to buck, the values of voltages and currents increase and so the errors increase as well, reaching the maximum value in the last working point in buck mode. However, because RMSE and MAE are not much above 1, it is possible to say that the model is enough accurate in its predictions, even if they are not completely exact. Also, it is possible to outline that the inductor current i_L has large values of RMSE in all the working points. This was expected, because the mathematical model is a state space averaging model, that is not able to account for the triangular shape of the inductor current and so also for the ripple that may be present on the other quantities. Also, the ESRs play an important role in defining the value of the inductor current: because they are neglected in this model, an important information is lost.

On the other hand, RAE and RSE can be used in order to compare the errors on the different signals, thus because they are not scale-dependent. The value of the RAE is important, because if it is greater than 1, it means that the model is poor. Observing the tables, it is possible to see that the model is not always very accurate: especially in boost and buck-boost there are one or more values that exceed 1. It is also interesting to notice that, for the pure buck mode, the RAE

and the RSE indicate that the model is quite accurate in predicting the values for all the signals of interest.

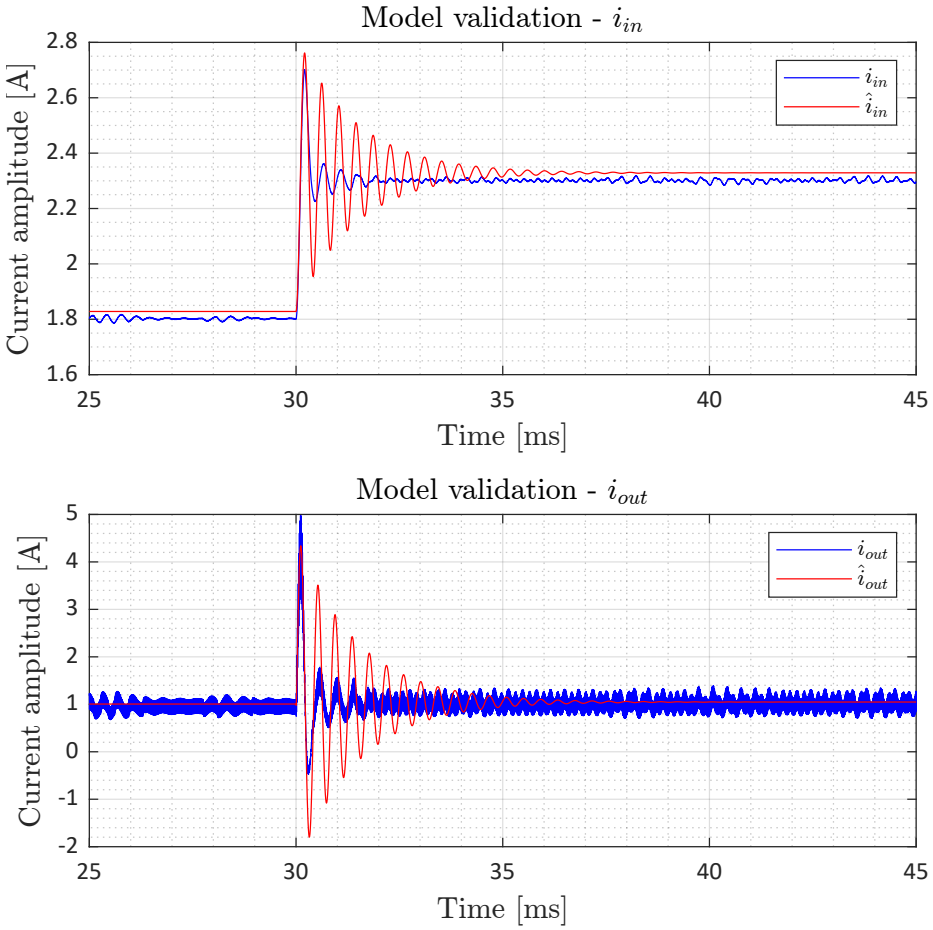


Figure 2.11: Comparison between i_{in} and $\hat{i}_{in}$ and between i_{out} and $\hat{i}_{out}$. Boost working point, with $v_{in} = 11\,\mathrm{V}$ and $u = 0.45$

Measure	MAE	RMSE	RAE	RSE
$v_{C_{in}}$	0.0921	0.1328	0.2021	0.0630
$v_{C_{out}}$	0.0045	0.0080	1.4269	1.5373
i_L	0.6780	0.9519	1.1365	1.2367
i_{in}	0.0384	0.0553	0.2021	0.0630
i_{out}	0.2009	0.3575	1.4296	1.5373

Table 2.6: Evaluation indices for boost working point, with $v_{in} = 11\,\mathrm{V}$ and $u = 0.45$

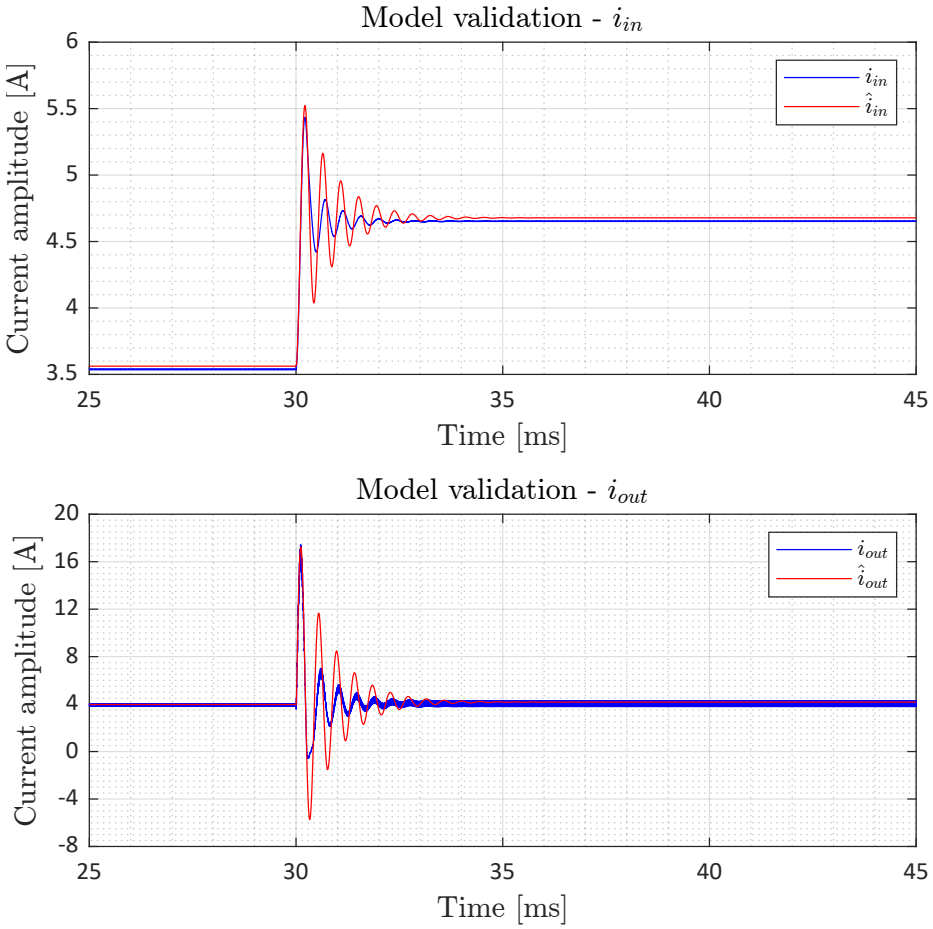


Figure 2.12: Comparison between i_{in} and $\hat{i}_{in}$ and between i_{out} and $\hat{i}_{out}$. Buck-boost boost working point, with $v_{in} = 22\,\mathrm{V}$ and $u = 0.052$

Measure	MAE	RMSE	RAE	RSE
$v_{C_{in}}$	0.0906	0.1563	0.0891	0.0176
$v_{C_{out}}$	0.0066	0.0180	0.9437	0.5739
i_L	0.4053	0.9631	0.7984	0.5397
i_{in}	0.0378	0.0651	0.0891	0.0176
i_{out}	0.2928	0.7987	0.9437	0.5739

Table 2.7: Evaluation indices for buck-boost boost working point, with $v_{in} = 22\,\mathrm{V}$ and $u = 0.052$

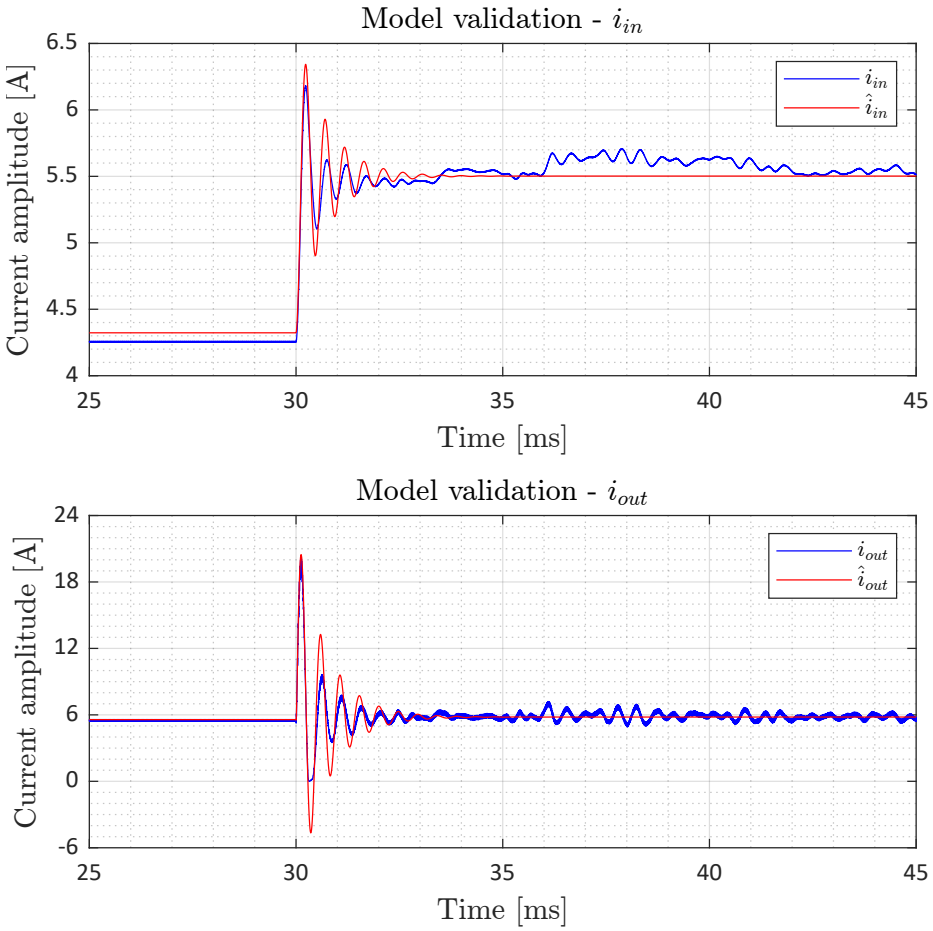


Figure 2.13: Comparison between i_{in} and $\hat{i}_{in}$ and between i_{out} and $\hat{i}_{out}$. Buck-boost buck working point, with $v_{in} = 26\,\mathrm{V}$ and $u = -0.024$

Measure	MAE	RMSE	RAE	RSE
$v_{C_{in}}$	0.1805	0.2188	0.1526	0.0255
$v_{C_{out}}$	0.0074	0.0151	0.6721	0.2725
i_L	0.4591	0.7842	0.7132	0.2841
i_{in}	0.0752	0.0912	0.1526	0.0255
i_{out}	0.3273	0.6712	0.6721	0.2725

Table 2.8: Evaluation indices for buck-boost buck working point, with $v_{in} = 26\,\mathrm{V}$ and $u = -0.024$

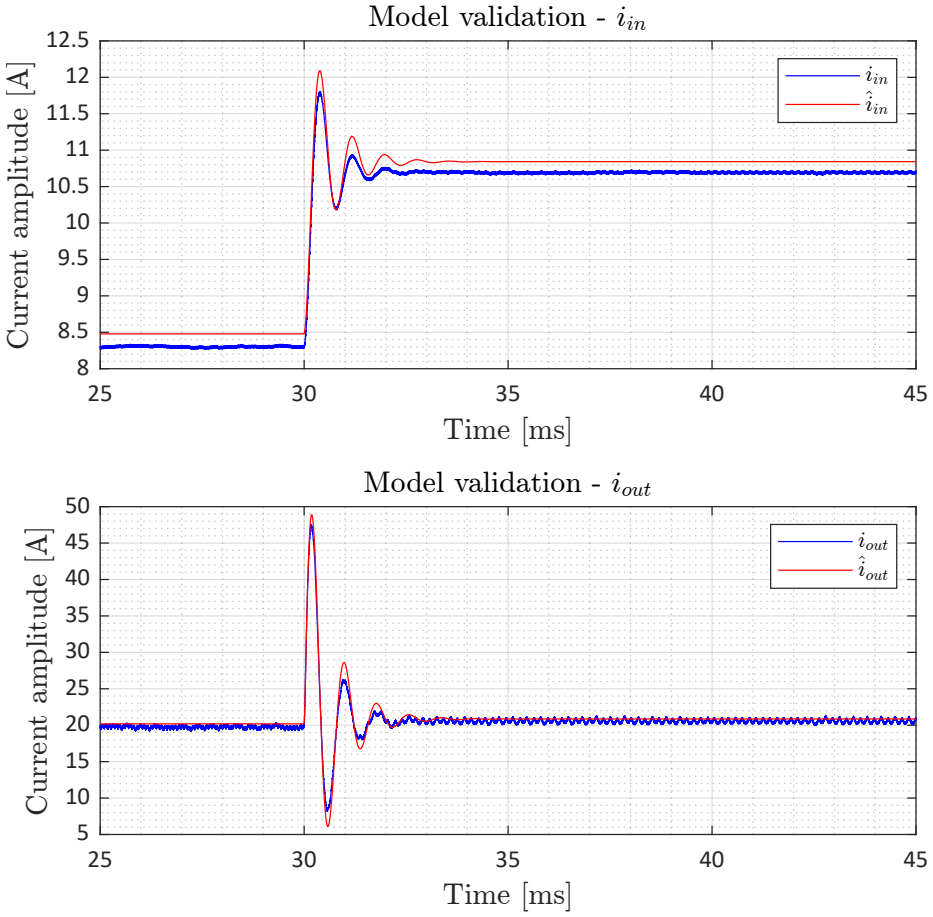


Figure 2.14: Comparison between i_{in} and $\hat{i}_{in}$ and between i_{out} and $\hat{i}_{out}$. Buck working point, with $v_{in} = 50\,\mathrm{V}$ and $u = -0.38$

Measure	MAE	RMSE	RAE	RSE
$v_{C_{in}}$	0.3801	0.3879	0.1736	0.0236
$v_{C_{out}}$	0.0100	0.0133	0.4671	0.0390
i_L	1.1456	1.3638	0.7043	0.1759
i_{in}	0.1584	0.1616	0.1736	0.0236
i_{out}	0.4461	0.5924	0.4671	0.0390

Table 2.9: Evaluation indices for buck working point, with $v_{in} = 50\,\mathrm{V}$ and $u = -0.38$

Finally, from the graphical analysis, a qualitative evaluation of the goodness of the model can be derived. Observing the figures 2.11 - 2.14, the model without the ESRs effect seems to be more accurate in buck mode than in boost mode. The evaluation is performed neglecting the first instants of time after the turn on, due to the presence of a current spike, related to the input capacitor C_{in}. A capacitor stores energy as a function of voltage and tries to keep the voltage constant over the time. Whenever the voltage suddenly changes (e.g. at the turn on of the converter), the capacitor reacts to this change supplying current to or drawing current from the voltage source. This is the cause of the current spike, that can distort the accuracy of the model evaluation.

As final remark, even if this model is not perfect at all, it can be used as good enough replacement of the model with the ESRs if necessary, allowing to perform the control design without representation issues.

Chapter 3

PI control design

The PI control is the the most common control solution applied in the industrial framework: as a matter of facts, it is very simple to be designed and it can be efficiently applied to every kind of system. This chapter outlines the design procedure and the capabilities of this type of control technique. At first, a brief introduction about the state of art is provided. Then the control design is carried out, both in Continuous Time (**CT**) and Discrete Time (**DT**), and its performances are evaluated through Simulink simulations.

3.1 State of art

In the literature, the applications of the PI and its variations for control DC-DC converters are numerous. In [14] and [15] the authors describe how to apply and design a PID controller for a buck DC-DC converter. [14] describes the common procedure that is applied for this type of DC-DC converters in terms of modeling (i.e. the state space averaging technique previously shown in chapter 2) and develops the design of the controller in discrete time, for a digital application. On the other hand, [15] pays more attention to the open loop analysis and the robust stability, developing and tuning the PID controller with three different techniques: using gain and phase margin specifications, pole-zero cancellation and with frequency loop shaping. Also, this research outlines that, applying frequency loop shaping, better performances and greater robustness can be reached.

In [16] a PI controller is designed for current control and applied to a specific variation of the buck-boost topology, called "versatile buck-boost converter". In this case, the PI is developed in discrete time, for digital implementation, and it is used for the control of the input current provided by an ideal voltage generator. The authors carry the design mainly focusing on boost mode, characterized by lowest crossover frequency and phase margin.

[17] develops an hybrid control system for digital implementation in a positive

buck-boost converter, based on a Digital Signal Processor controller. The converter is controlled using two different DT PI controllers, implementing two different control loops: an inner control loop on the inductor current and an outer control loop on the output voltage, combining the advantages of current and voltage control modes.

In [18] a variation of the PID is implemented, that allows to obtain efficient tracking and good load disturbance regulation. This variation is called Pseudo-derivative Feedback with Feed-forward, that can be seen as a generalization of the PID controller: this is a two-degree-of-freedom controller, in which performance tracking and load disturbances are tuned separately.

Other authors try to improve the performances of the PID controller exploiting "smart" methods of tuning. As example, [19] and [20] implement matheuristic algorithms in order to tune the parameters of the controller in an optimal way. In particular, [19] develops the control design of a PI converter, applying the Ant Colony Optimization (**ACO**) algorithm, in order to optimally choose the PID parameters. Instead, [20] starts the design from a variation of the PID controller, the Fractional-Order PID, that consists of five parameters, instead of the three parameters used for the classical PID. Then, the tuning is performed using the Particle Swarm Optimization (**PSO**) algorithm, that provides an optimal choice in regards of the necessary requirements.[21] on the other hand, try to overcome the problems of the matheuristic algorithms (weak local search, slow convergence rate and trapping in local optima) leaving the tuning of the parameters to a trained neural networks. In this work, the authors use the PSO in order to evaluate optimal values for the weights of the neural network, which is used for implementing the mathematical law of the PID controller.

3.2 Introduction

In this section, the design of a classical PI controller is introduced. The design procedure is carried taking into account the gains and the phase margins of the system and trying to obtain a crossover frequency as high as possible (i.e. to improve the response speed of the system itself). Referring to what is written in the second chapter, the input current i_{in} is used as control variable. This implies the realization of an output-feedback control, that is an outer current loop, as depicted in figure 3.1.

There is a total of four inputs and two outputs. v_{in} and v_{out} are the electrical inputs, which set the working point of the plant. d_a and d_b, instead, are the command inputs, that are related to the control command u through the dual carrier modulation. The control variable i_{in} is compared with the reference signal i_{ref}, imposed as the needed current for obtaining the maximum power transfer.

Considering the maximum power transfer theorem, the load voltage has to be

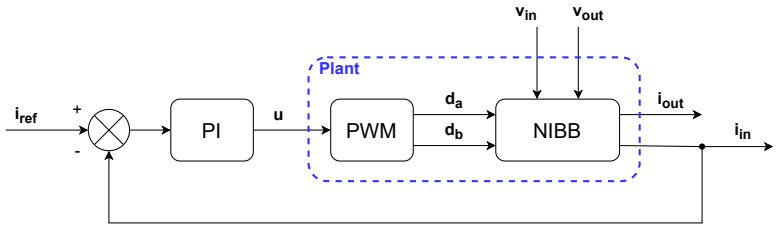


Figure 3.1: General block diagram of the control system with PI controller

equal to one-half of the Thevenin voltage equivalent of the source. So, for the NIBB converter:

$$v_{C_{in}} = \frac{v_{in}}{2} \implies i_{in} = \frac{v_{in}}{2R_{in}} \tag{3.1}$$

According to (3.1), the reference current changes depending on the working point and it is defined as:

$$i_{ref} = \frac{v_{in}}{2R_{in}}$$

In order to design a PI controller, it is necessary to consider a series of different working points. As a matter of facts, the PI is a linear controller, while the plant is a non-linear system. The values of the parameters of the transfer functions of i_{in} with respect to d_a and d_b depend on the values of the big signals and so on the working condition of the system itself. This implies that the support of the design procedure is not only one transfer function but a set of transfer functions.

W.P.	D_a	D_b	V_{in} [V]	V_{out} [V]
A	1	0.6421	8	12
B	1	0.3831	15	12
C	0.6204	0	40	12
D	0.4314	0	60	12

Table 3.1: Working points, with related duty cycles and voltage values

A possible choice can be a set of four transfer functions, evaluated in four different working points: two in the boost area and two in the buck area. The buck-boost area can be neglected, assuming that the behavior in buck-boost boost mode can be assimilated to a simple boost mode and the behavior of the circuit in buck-boost buck mode can be assimilated to a simple buck mode. These points are the same defined in table 2.4, that are reported again here, in table 3.1, only for clarity sake.

Moreover, because the working points are set and no symbolical representation is needed, the model with the ESRs can be used in order to compute the necessary transfer functions, with greater accuracy.

3.3 Continuous time design

For the design of the controller, the structure exploited is a simple PI, without the introduction of a derivative gain, that is unnecessary for this application. The control transfer function in Laplace domain, namely $C(s)$, is shown in equation (3.2), with figure 3.2 that depicts the block diagram of the controller.

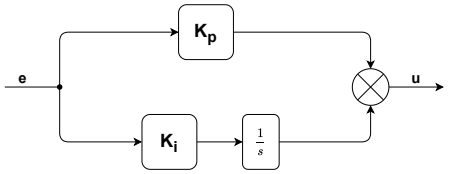


Figure 3.2: Block diagram of the PI controller in CT

$$C(s) = K_p + \frac{K_i}{s} = K_p \cdot \frac{s + \frac{K_i}{K_p}}{s} \tag{3.2}$$

The first step is to determine the sign of the gain K_p: the Nyquist diagrams of the working points are very useful in this sense. Taking into account the four working points specified in table 3.1, the transfer functions obtained are the following:

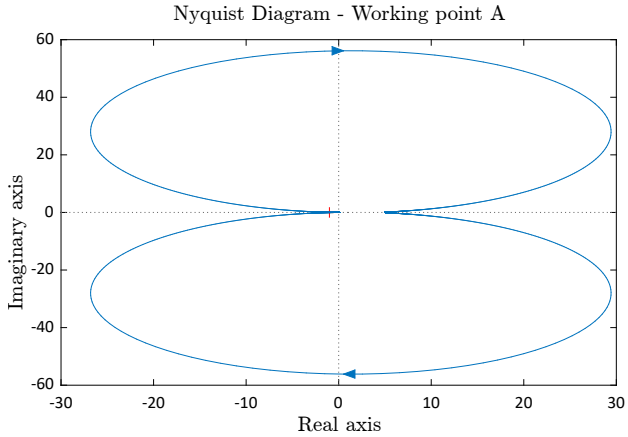


Figure 3.3: Nyquist diagram of NIBB transfer function in working point A

$$G_A = \frac{200.18(s + 5.721e06)(s + 1e05)}{(s + 9.965e04)(s^2 + 1396s + 2.298e08)}$$

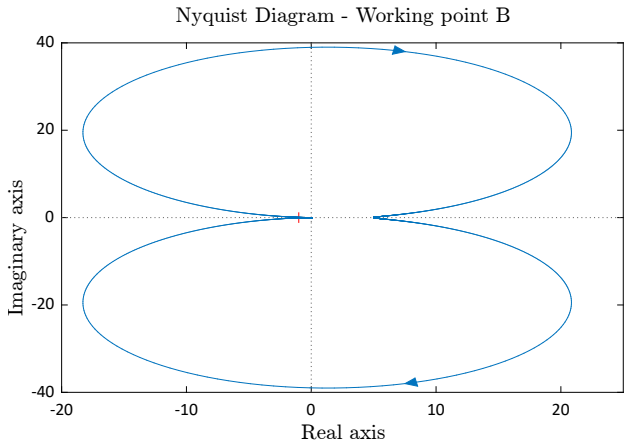


Figure 3.4: Nyquist diagram of NIBB transfer function in working point B

$$G_B = \frac{200.7(s + 5.721e06)(s + 1.003e05)}{(s + 9.91e04)(s^2 + 1959s + 2.317e08)}$$

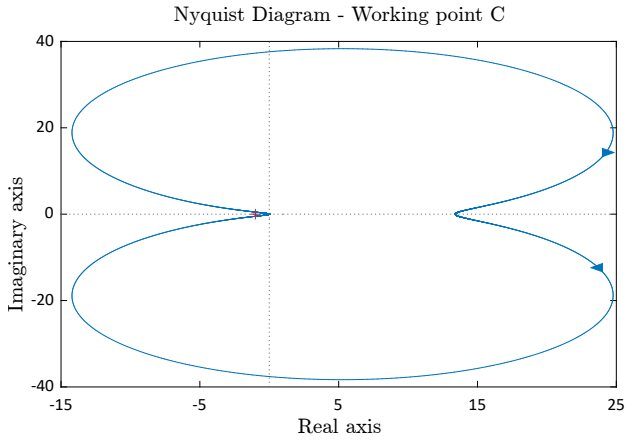


Figure 3.5: Nyquist diagram of NIBB transfer function in working point C

$$G_C = \frac{0.0022312(s + 5.721e06)(s^2 + 1.928e05s + 9.498e09)}{(s + 9.768e04)(s^2 + 3373s + 9.241e07)}$$

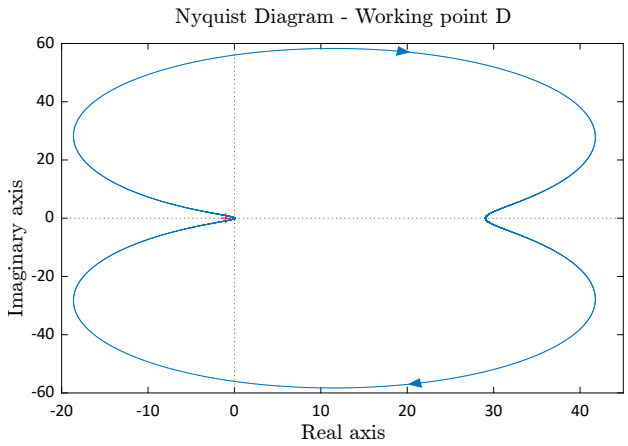


Figure 3.6: Nyquist diagram of NIBB transfer function in working point D

$$G_D = \frac{0.0048389(s + 5.721e06)(s + 9.558e04)(s + 4.907e04)}{(s + 9.767e04)(s^2 + 3376s + 4.587e07)}$$

Observing the Nyquist diagrams of the four points, no one of the diagrams encircles the critical point (-1,0). Moreover, in all the transfer functions of the plant for the different working points there are no unstable poles. With these information, it is possible to correctly define the value of K_p that does not lead to instability. Indeed, considering a closed loop system where the controller is defined only by the gain K_p, the position of the encirclements on the Nyquist diagram changes depending on the value of K_p itself. The set of inequalities (3.3) - (3.6) show the constraints in the choice of K_p for guarantying stability of the closed loop system:

$$K_{p_A} > 0 \quad \text{and} \quad K_{p_A} \in [-0.2\ ;\ 0] \tag{3.3}$$
$$K_{p_B} > 0 \quad \text{and} \quad K_{p_B} \in [-0.2\ ;\ 0] \tag{3.4}$$
$$K_{p_C} > 0 \quad \text{and} \quad K_{p_C} \in [-0.0746\ ;\ 0] \tag{3.5}$$
$$K_{p_D} > 0 \quad \text{and} \quad K_{p_D} \in [-0.0344\ ;\ 0] \tag{3.6}$$

A positive value of K_p can guarantee the stability of the system in all the working points. Also, from the Nyquist analysis, it is possible to see that K_p can assume small negative values inside a well defined range. However, because this range is different for each point and because to shift the phase of 180 deg could lead to stability problems of the final controller, it is better to choose a positive value of K_p. Thus, also because there is no upper limit in the positive range of K_p and so there is a greater freedom for the value choice.

The tuning of K_p is performed aiming to stability and robustness, using the evaluation of phase and gain margins. The idea is to have a gain margin greater

than 8 dB and a phase margin greater than 60 deg, in order to be sure that the designed controller is robust.

About K_i, its value is chosen in order to compensate the transfer function of the system so that the open loop transfer function behaves like an integrator with a certain crossover frequency (i.e. a null tracking error for a step reference is desired). As a matter of facts, the PI introduces a zero with frequency:

$$f_z = \frac{K_i}{K_p}$$

Depending on the value of f_z, the crossover frequency changes and the system may become more responsive (i.e. fast). However, the faster is the system the less are the margins and so the robustness. The values of K_p and K_i that guarantee stability, good performances and good margins are:

$$K_p = 0.01 \quad \text{and} \quad K_i = 59$$

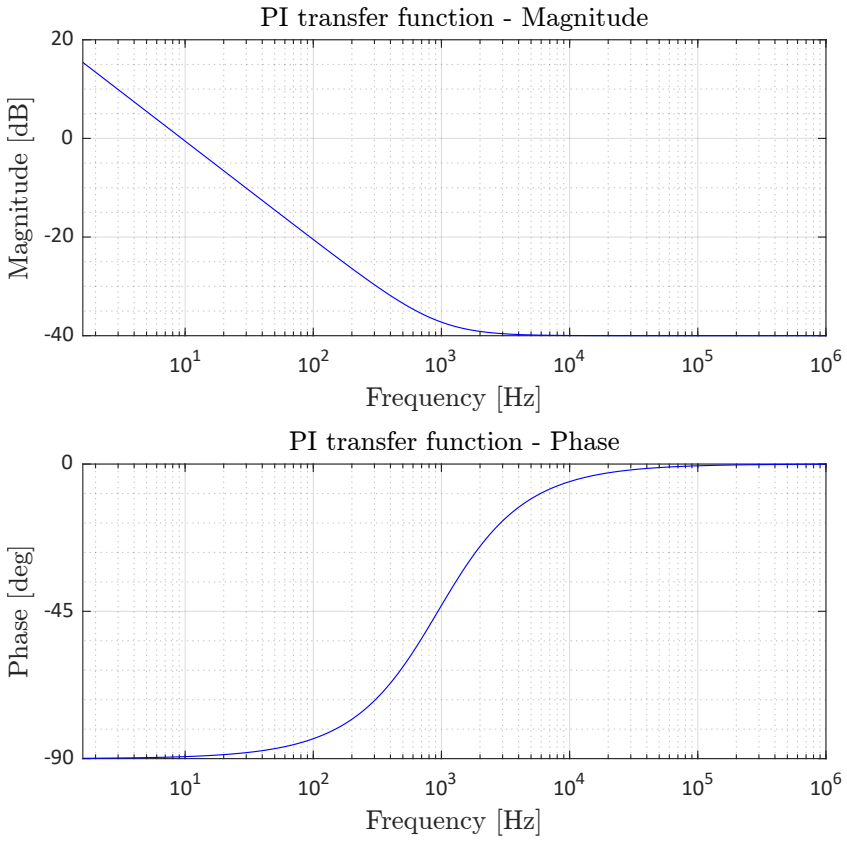


Figure 3.7: Bode diagram of the PI controller transfer function

3.3.1 Results

The gain margin γ, the phase margin ϕ and the crossover frequency f_c are listed in table 3.2. Figure 3.8 depicts the behavior of the open loop transfer function obtained applying the PI controller.

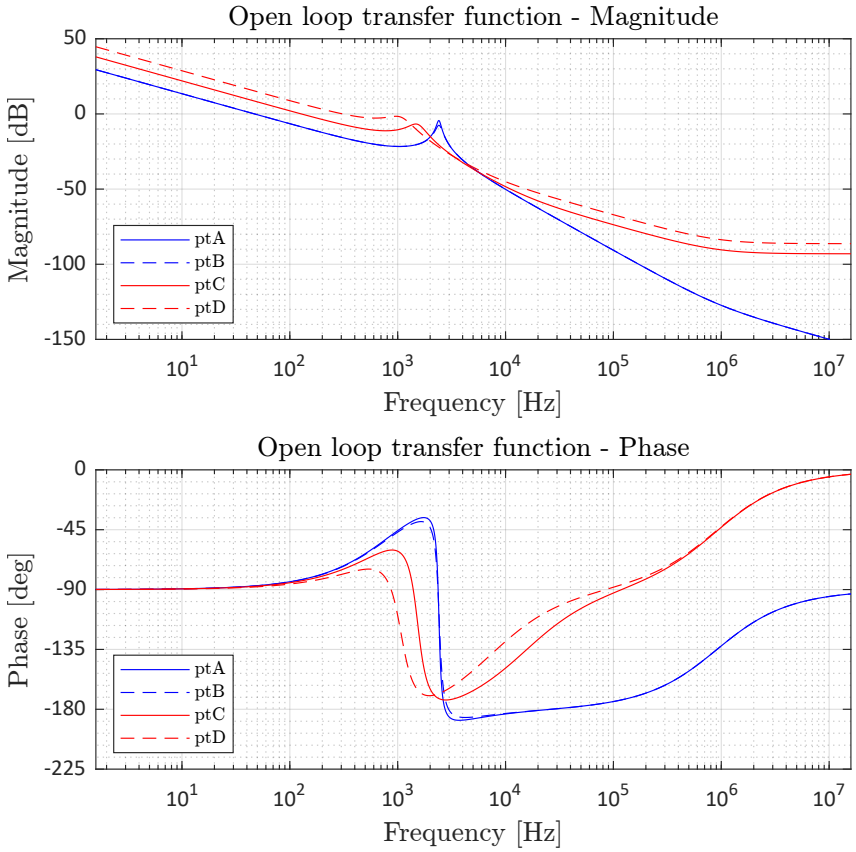


Figure 3.8: Bode diagrams of the NIBB open loop transfer functions, evaluated in four different working points in CT

W.P.	γ [dB]	ϕ [deg]	f_c [Hz]
A	15.6	92.8	47.11
B	20.0	92.7	47.11
C	∞	96.5	127.80
D	∞	102.0	310.35

Table 3.2: Margins and crossover frequencies related to the PI controller, evaluated in four different working points

In order to test the time performances of the designed PI controller, a simulation is performed, using the Simulink model of the circuit, as previously described in chapter 2. The test is performed using a step function with unitary amplitude as reference signal. In order to neglect the behavior of the input capacitor at the turn on of the circuit, the step function starts from a value that is 1 A below the reference value (defined by the maximum power transfer theorem), maintaining the input and the output voltages constant. Then, after 40 ms the step is triggered and reaches the reference value. In this way, at first, the circuit is stabilized around a working condition and then the true reference is applied, allowing to check the performances of the closed loop system without distortions.

The simulation results are presented in figure 3.9 - 3.12. Table 3.3 resumes the obtained performances in terms of percentage overshoot ($\hat{s}_{\%}$), rise time (t_r), defined as the time value at which the signal reaches the 90% of the reference, and settling time at 3% ($t_{s,3\%}$).

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	0.82	8.46	11.47
B	0.28	7.76	8.38
C	0.15	2.81	1.58
D	0.25	1.22	0.36

Table 3.3: Values of overshoot, rise time and settling time in each working point of interest for PI controller in CT

Table 3.3 shows that the rise time decreases going from boost to buck. This is strictly related to the crossover frequency of the system, that, indeed, increases going from boost to buck. Moreover, the settling time behaves in a similar way, indicating that the buck working mode is more responsive than the boost working mode. However, both the rise time and the settling time are in the order of the ms, generally defining a very responsive system.

About the overshoot, it is very close to 0%: for this application, to have an overshoot at least less than 2% is fundamental, in order to avoid too large oscillations around the maximum power point.

Moreover, analyzing the margins, it is possible to say that the controller is very robust. Indeed, the phase margin for each open loop transfer function is very large and it is always greater than 60 deg.

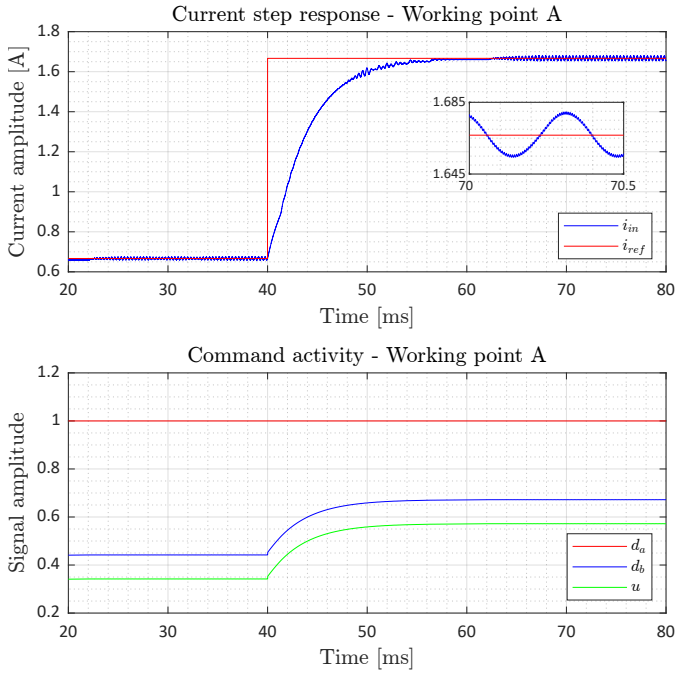


Figure 3.9: Simulation in working point A (boost) with PI controller in CT

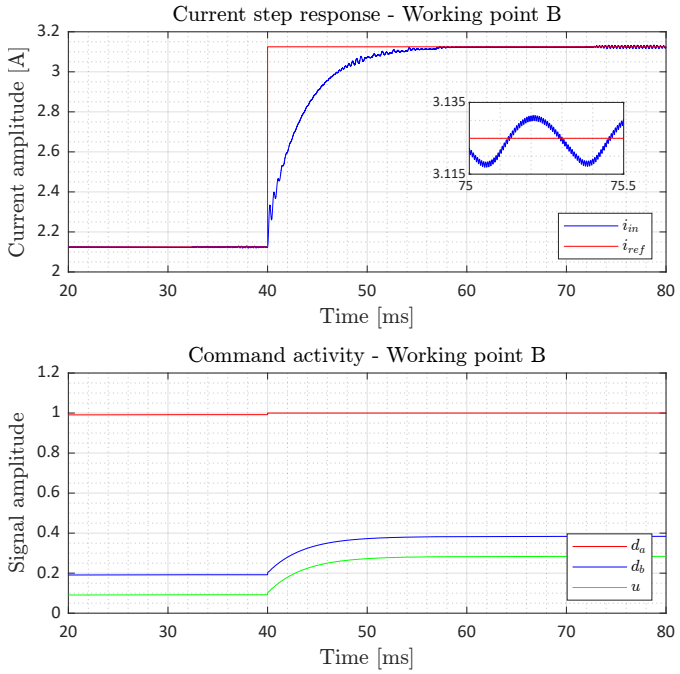


Figure 3.10: Simulation in working point B (boost) with PI controller in CT

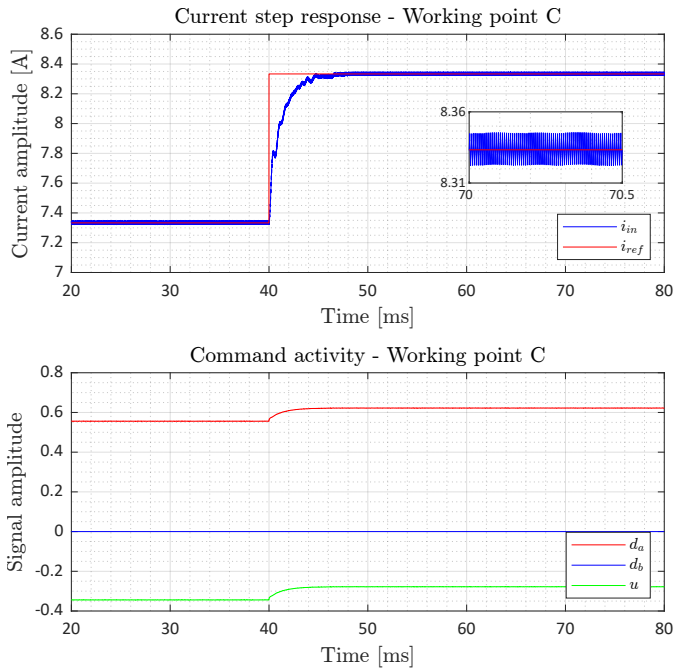


Figure 3.11: Simulation in working point C (buck) with PI controller in CT

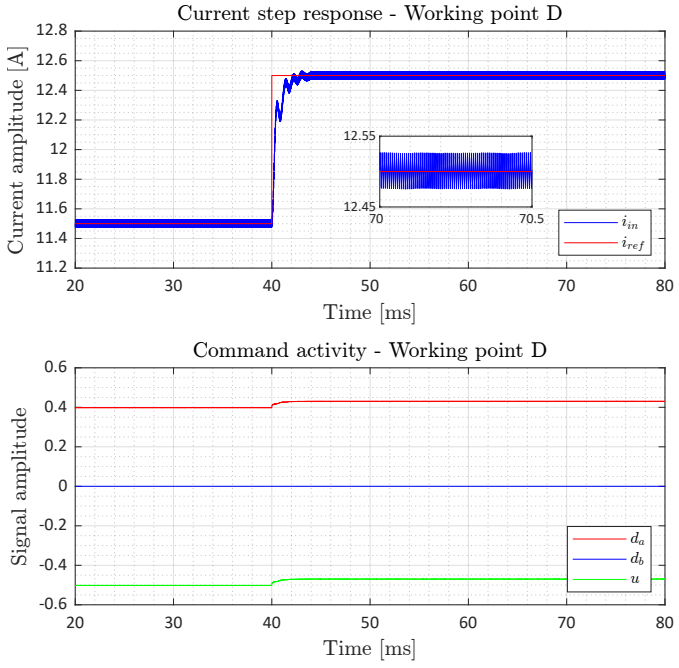


Figure 3.12: Simulation in working point D (buck) with PI controller in CT

3.4 Discrete time design

The design of the PI controller in discrete time is simple, since it is possible to define the corresponding discrete time structure, as written in (3.7)).

$$C(z) = K_p + K_i \cdot \frac{T_s}{z-1} \tag{3.7}$$

The values of the control parameters are equal to the ones chosen for the continuous case. T_s is the sampling time and its choice depends mainly on the working frequency of the controller in the hardware implementation. T_s is set as equal to $T_s = \frac{1}{f_s}$ with the sampling frequency f_s equal to 30 kHz. The controller, as formulated in (3.7), has the same behavior of the PI controller in CT.

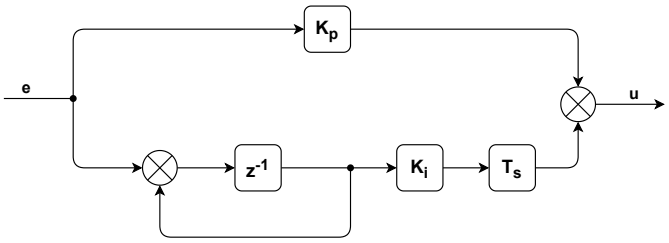


Figure 3.13: Block diagram of the PI controller in discrete time

3.4.1 Results

The results from the simulation are close to the ones obtained for the CT controller. Rise time, settling time and percentage overshoot are listed in table 3.4; figures 3.14 - 3.17 show the plots obtained from the simulations.

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	0.89	8.55	11.39
B	0.30	7.83	8.06
C	0.24	2.69	1.60
D	0.28	1.20	0.36

Table 3.4: Values of overshoot, rise time and settling time in each working point of interest for PI controller in DT

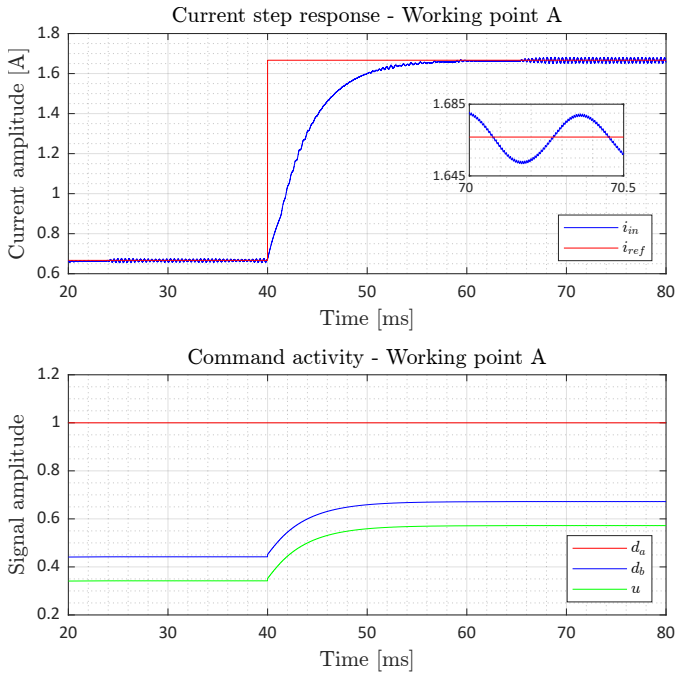


Figure 3.14: Simulation in working point A (boost) with PI controller in DT

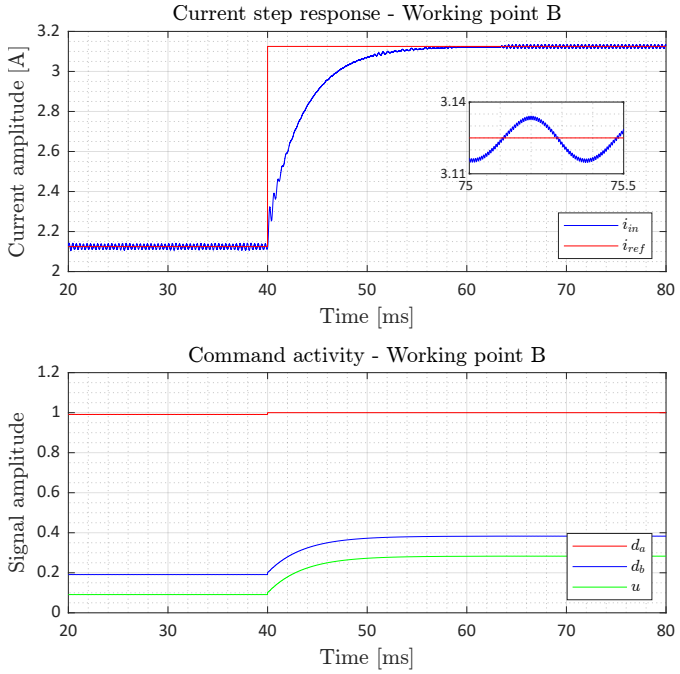


Figure 3.15: Simulation in working point B (boost) with PI controller in DT

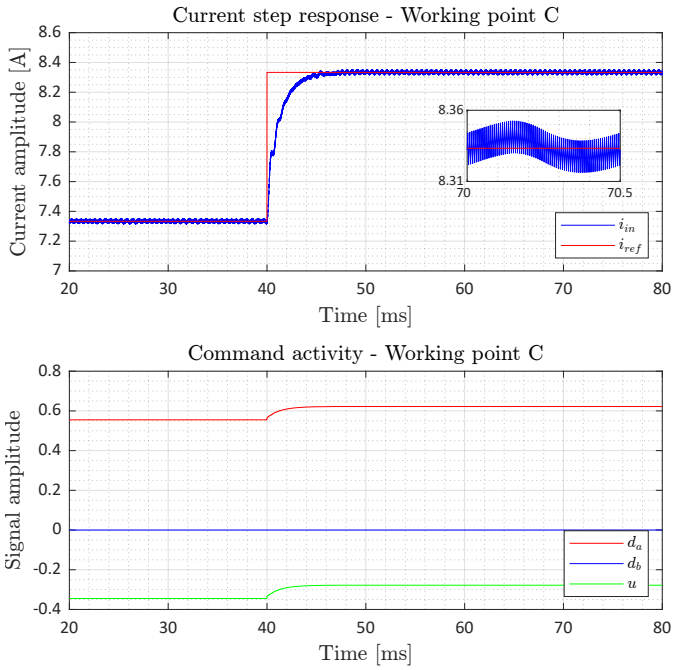


Figure 3.16: Simulation in working point C (buck) with PI controller in DT

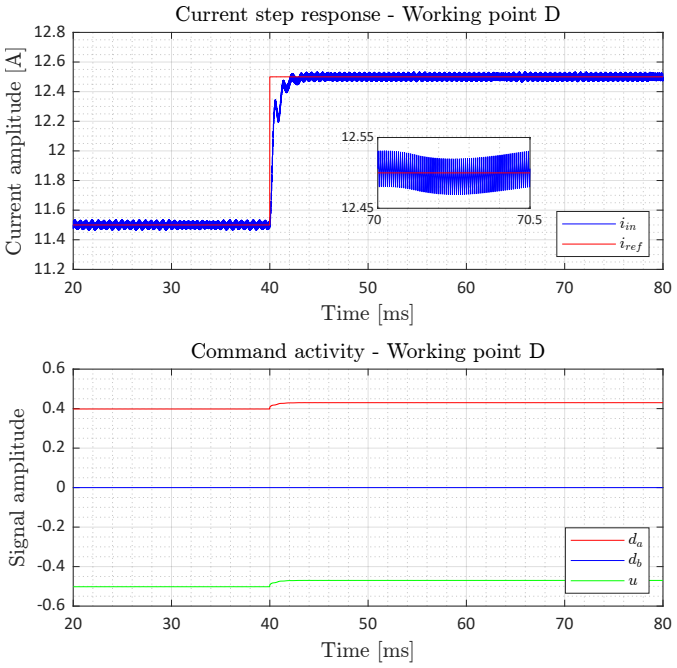


Figure 3.17: Simulation in working point D (buck) with PI controller in DT

Chapter 4

H-infinity control design

The PI introduced in chapter 3 is a simple linear controller, in which the tuning can be obtained varying two quantities, related to a proportional contribute and an integrative contribute. With a trial and error procedure, the controller parameters are modified until the desired requirements are achieved. However, there exist other control techniques that are able to construct an effective linear controller, including into the design the requirements related to the robustness and the desired performances. In this chapter, after a brief introduction of the related works present in the literature, the design of a linear controller is performed, exploiting the H_∞ technique. Then the controller transfer function is also translated in discrete time, using the emulation procedure for taking into account the contributes of the zero-order-hold (**ZOH**) filter and of the A/D converter.

4.1 State of art

Even if the PI seems to be the most common solution for the control of the DC-DC converters, other types of linear control strategies can be used in order to enhance the performance of the system and to account for other requirements or constraints.

[11] and [22] compare the performances obtained using a PID controller whit a type III compensator. The type III compensator allows to obtain a greater bandwidth, canceling the effect of the double complex poles that delay the system. Therefore, the design of the compensator aims to adjust the margins and to increase the response velocity of the system. [23] develops a quite similar idea, using instead a controller transfer function that is not proper: i.e the function has two zeros, that compensate a couple of complex poles, and one pole in the origin, that guarantees the zero tracking error requirement.

[24] and [25] exploit a set of Linear Matrix Inequalities (**LMIs**) in order to design a controller able to take into account for multiple requirements. [24] implements a controller for a boost DC-DC converter in which the design consider performance,

stability and control efforts constraints, implemented as LMIs. [25] develops a similar design for a modular buck-boost converter, in which the design procedure becomes an optimization problem, expressed with LMIs and subject to constraints on stability, pole placement and control effort.

4.2 Introduction

The H_∞ design is a complex design technique that can be used in order to build a linear controller, able to ensure better performances than the ones achieved with the PI controller. In H_∞ design, the control transfer function is obtained solving the optimization problem written in equation (4.1).

$$C(s) = \arg \min_{C \in C^{stab}} \|T_{wz}(s)\|_\infty \tag{4.1}$$

The solution can be provided expressing the constraints in the form of LMIs. This approach is based on a state space representation of the generalized plant M, that can be described by the transfer function T_{wz},

$$T_{wz}(s) = \begin{bmatrix} W_1(s)S_n(s) \\ W_2(s)T_n(s) \end{bmatrix} \tag{4.2}$$

T_{wz} expresses the constraints applied to the problem through suitable functions called "weighting functions", W_1 and W_2: they are transfer functions in the Laplace domain that embody the constraints/requirements required for the control design. Taking into account T_{wz}, the generalized plant M is built as shown in figure 4.1.

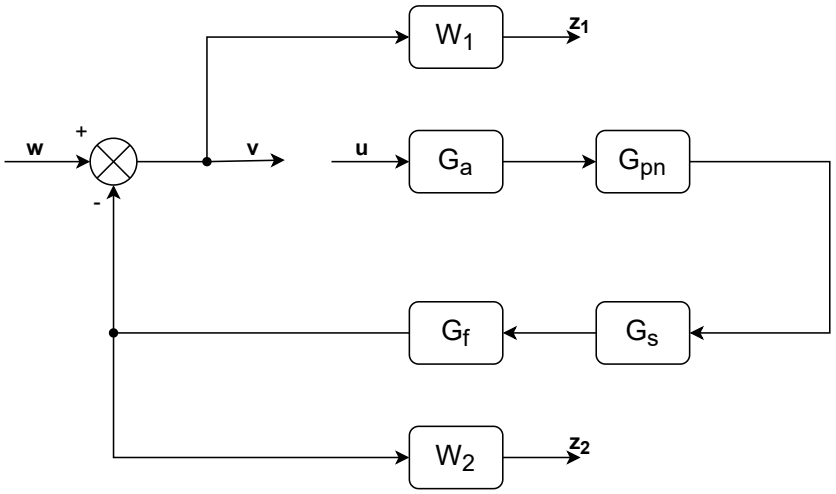


Figure 4.1: Block diagram of the generalized plant M

with:

- G_a = actuator transfer function;
- G_{pn} = nominal plant transfer function;
- G_s = sensor transfer function;
- G_f = filter transfer function;

v and u in 4.1 are, respectively, the input and the output signals of the controller C. The choice of the weighting functions depends on which requirements have to be satisfied: with this design technique is possible to take into account both the nominal performance and the robust stability conditions.

The nominal performance refers to the conditions that the nominal function has to respect, expressed as time requirements that are then translated in the frequency domain. This procedure can be performed using weighting functions that define the shape of the sensitivity function S_n and of the complementary sensitivity function T_n of the nominal plant. Referring to the figure 4.1, S_n and T_n can be built as:

$$\begin{aligned} \text{Open-Loop: } L_n(s) &= C(s)G_aG_{pn}(s)G_sG_f \\ \text{Sensitivity: } S_n(s) &= \frac{1}{1+L_n(s)} \\ \text{Comp. sensitivity: } T_n(s) &= \frac{L_n(s)}{1+L_n(s)} \end{aligned}$$

The requirements on S_n are translated in the frequency domain with the weighting function W_s, while the requirements on T_n are expressed with the weighting function W_t. However, the nominal function does not take into account for the uncertainty of the plant. This is important, because the controller has to work without leading to instability the real plant, that is not totally described by the nominal transfer function. For this reason, it is necessary another suitable weighting function that takes into account for the robust stability condition. This function is called W_u and it can be defined in different ways depending on the uncertainty of the model set chosen. Then the controller transfer function is computed such that the inequalities of (4.3) are satisfied.

$$\|W_s(s)S_n(s)\|_\infty < 1 \qquad \|W_t(s)T_n(s)\|_\infty < 1 \qquad \|W_u(s)T_n(s)\|_\infty < 1 \tag{4.3}$$

These inequalities represent sufficient (but not necessary) conditions that ensure the achievement of the requirements on nominal performance and robust stability.

However, it is always necessary a check in the time domain in order to be sure that the designed controller respects the desired behavior. Recalling what written about T_{wz}, W_1 is chosen as equal to W_s, while W_2 is set in order to take into account for both robust stability and nominal performance on T_n:

$$|W_2(j\omega)| = \max(|W_u(j\omega)|, |W_t(j\omega)|)$$

In order to build the controller with H_∞ design, it is necessary to solve the optimization problem stated in equation (4.1). The approach used is LMI-based, a very handful approach, also because it is directly implemented in the Matlab LMI control system toolbox. The LMI procedure is based on the state space description of the generalized plant M, as expressed in (4.4).

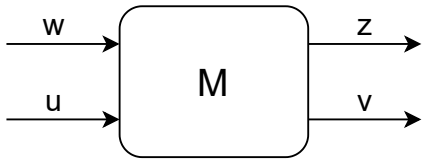


Figure 4.2: Schematic representation of M block

$$M : \begin{cases} \dot{x}_M(t) = Ax_M(t) + B_1 w(t) + B_2 u(t) \\ \dot{z}(t) = C_1 x_M(t) + D_{11} w(t) + D_{12} u(t) \\ \dot{v}(t) = C_2 x_M(t) + D_{21} w(t) + D_{22} u(t) \end{cases} \tag{4.4}$$

x_M is defined as the union of the state variables of the nominal model G_{pn} and of the weighting functions W_1 and W_2. Then, the LMI optimization problem can be solved under the following assumptions:

1. the matrix triplet (A,B_2,C_2) is stabilizable and detectable;
2. $D_{22} = 0$

About assumption 1, it is necessary to have all the eigenvalues of unobservable and uncontrollable part of the system stable. Therefore, the generalized plant M can be internally stabilized by a Linear Time Invariant (**LTI**) controller C if and only if W_1 and W_2 are stable transfer functions. This is an important result that will be taken into account in the definition of the weighting functions.

Assumption 2, instead, only implies that the nominal transfer function of the plant has to be strictly proper. The transfer functions of the NIBB in the different working points are, in general, proper, so a small modification is needed in order to apply this type of design technique. A more detailed description about the H_∞ design is outlined in [26].

As final remark, the formulation of the H_∞ design with LMI optimization is based on the so called Bounded Real Lemma. Considering a minimal state space realization (A_{wz}, B_{wz}, C_{wz}, D_{wz}) of the transfer function $T_{wz}(s)$, the stabilizing LTI controller $C(s)$ which minimizes the H_∞ norm of $T_{wz}(s)$ is given by:

$$
\begin{aligned}
&C(s) = \arg\ \min\ \gamma \\
&\text{s.t.} \\
&P = P^T \geq 0 \\
&\begin{pmatrix} A_{wz}^T P + P A_{wz} + C_{wz} C_{wz}^T & P B_{wz} + C_{wz}^T D_{wz} \\ B_{wz}^T P + D_{wz}^T C & D_{wz}^T D_{wz} - \gamma^2 I \end{pmatrix} \leq 0
\end{aligned}
$$

4.3 Continuous time design

The application of the H_∞ design to the control problem of the NIBB converter is based on a particular modification of the plant and of the uncertainty set that is necessary to take into account. The uncertainty set can be defined as the set of transfer functions (i.e. models) that can describe the behavior of the plant. It takes into account the type of uncertainty (dynamic or parametric) and it is used in the design in order to obtain a robust controller. For the NIBB converter, the transfer function changes depending on the working point, so there is not a unique representation of the system. The basic idea is so to consider the "uncertainty about the working point" as part of the uncertainty set. This implies the construction of a set of transfer functions that represents all the possible conditions of the plant (see figure 4.3). Each transfer function is obtained changing the voltage v_{in} from 8 to 60 V with steps of 1 V and applying maximum power transfer condition. In order to apply the H_∞ design technique, one transfer function has to be chosen as nominal: the most suitable candidate is the highest buck transfer function (with $v_{in} = 60\,\text{V}$, equation (4.5)). The buck mode is characterized by a more responsive transfer function and using the highest buck ensures to avoid a too fast closed loop system that could be unstable.

$$
G_p(s) = \frac{0.0049192(s + 5.721e06)(s + 9.572e04)(s + 4.738e04)}{(s + 9.767e04)(s^2 + 3376s + 4.564e07)} \tag{4.5}
$$

However, this type of transfer function is not strictly proper, so a modification is needed in order to continue with the design. This modification consists in simplifying poles and zeros with close frequencies and neglecting zeros at high frequency. The result is shown in equation (4.6).

$$
G_{pn}(s) = \frac{1.3068e09}{s^2 + 3376s + 4.564e07} \tag{4.6}
$$

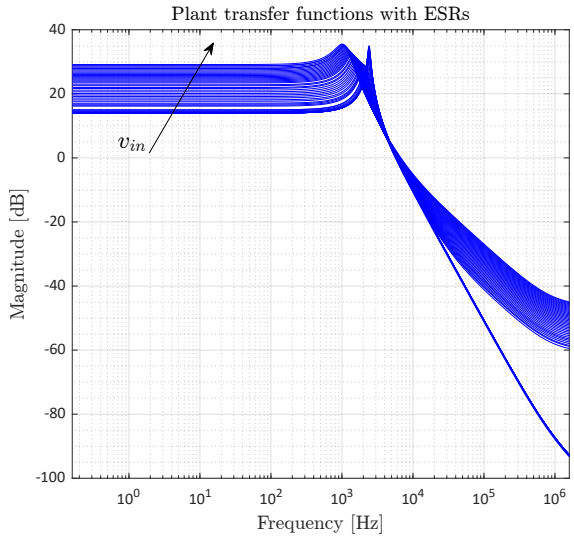


Figure 4.3: Transfer functions of 53 working points in the range $v_{in} \in [8\ 60]\,\mathrm{V}$

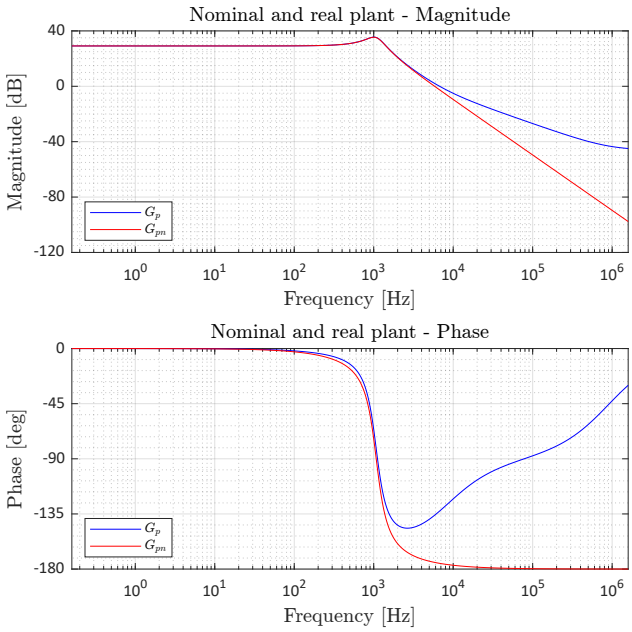


Figure 4.4: Graphical comparison of $G_{pn}(s)$ with $G_p(s)$ for CT design

Then, a set of desired time requirements is taken into account:

1. unitary scaling factor $K_d = 1$;
2. zero tracking error due to unitary step reference;
3. overshoot $\hat{s}_{\%} \leq \hat{s}_0 = 1\%$;
4. rise time $t_r \leq t_{r0} = 1.5\,\text{ms}$;
5. settling time at 3% $t_{s,3\%} \leq t_{s0} = 1\,\text{ms}$.

The first requirement defines the ratio between the desired output and the reference and it is fundamental in order to set the filter gain G_f. Considering the second requirement, the tracking error is defined as:

$$e_r^\infty = \lim_{t\to\infty} e_r(t) = 0 \quad \text{with} \quad e_r(t) = K_d r(t) - y(t) \tag{4.7}$$

Because the second requirement implies that the system type has to be, at least, equal to 1 (i.e. one pole in zero in the controller is needed), from equation (4.7) it is possible to compute the filter gain, as written in equation (4.8). Assuming G_a and G_s as constant unitary gains, G_f can be set so as equal to 1.

$$K_d = \frac{1}{G_f G_s} \implies G_f = \frac{1}{K_d G_s} \tag{4.8}$$

Then, requirements from 3 to 5 are translated in frequency domain. Requirement 3 defines constraints on the resonance peak of the complementary sensitivity function (T_p) and of the sensitivity function (S_p).

$$T_p \leq \frac{1}{2\zeta\sqrt{1-\zeta^2}} = T_{p0} \qquad S_p \leq \frac{2\zeta\sqrt{2+4\zeta^2+2\sqrt{1+8\zeta^2}}}{\sqrt{1+8\zeta^2+4\zeta^2-1}} = S_{p0}$$

with the damping factor, ζ, defined as a function of the overshoot.

$$\zeta \geq \frac{|\ln(\hat{s}_0)|}{\sqrt{\pi^2 + \ln^2(\hat{s}_0)}}$$

Requirements 4 and 5 introduce constraints on the natural frequency ω_n and on the crossover frequency ω_c.

$$\omega_{n,4} \geq \frac{\pi - \arccos\zeta}{t_{r0}\sqrt{1-\zeta^2}} \qquad \omega_{n,5} \geq -\frac{\ln(\alpha/100)}{t_{s0}\zeta} \qquad \omega_c \geq \omega_n\sqrt{\sqrt{1+4\zeta^4} - 2\zeta^2}$$

Taking into account all the constraints introduced, table 4.1 resumes all the numerical values chosen for the design.

Quantity	Numerical value
ζ	0.8261
T_{p0}	1.0740
S_{p0}	1.2117
ω_n	4244.8
ω_c	2427.9

Table 4.1: Numerical values of the quantities used for the H_∞ design

With all the information recollected until now, it is possible to build the two weighting functions W_s and W_t. The first weighting function, W_s is built using the requirements expressed by S_{p0}, ζ and ω_n. S_{p0} is an upper limit that the sensitivity function has to respect, while ζ and ω_n are represented through a prototype sensitivity function S^{II}.

$$S^{II}(s) = \frac{s(s + 2\zeta\omega_n)}{s^2 + 2 * \zeta\omega_n s + \omega_n^2}$$

The weighting function is built starting from its inverse, using Butterworth polynomials. From (4.3) is possible to obtain:

$$|S_n(j\omega)| \leq \left|W_s^{-1}(j\omega)\right| \quad |T_n(j\omega)| \leq \left|W_t^{-1}(j\omega)\right| \quad \forall\omega$$

In general, to build the inverse of the weighting function is easier, because a graphical representation can be used in order to tune the function parameters. W_s^{-1} can be chosen as:

$$W_s^{-1}(s) = \frac{a(s^{v+p})(1 + s/\omega_1)}{1 + 1.414s/\omega_2 + (s/\omega_2)^2}$$

where $v + p$ is the type of the system, that in this case has to be equal to 1 (i.e. $p = 0$ is the number of poles in zero of the plant, while $v = 1$ is the number of poles in zero of the controller). a and ω_1 are free parameters, while ω_2 is set imposing that, at high frequencies, the transfer function shall follow the limit defined by S_{p0}.

$$\lim_{s\to\infty} W_s^{-1}(s) = \frac{a\omega_2^2}{\omega_1} = S_{p0} \implies \omega_2 = \sqrt{\frac{S_{p0}\omega_1}{a}}$$

a and ω_1 are tuned in order to stay close to the prototype sensitivity function. The graphical result is shown in figure 4.5. Transfer functions W_s^{-1} and S^{II} are then written in equations (4.9) and (4.10) respectively.

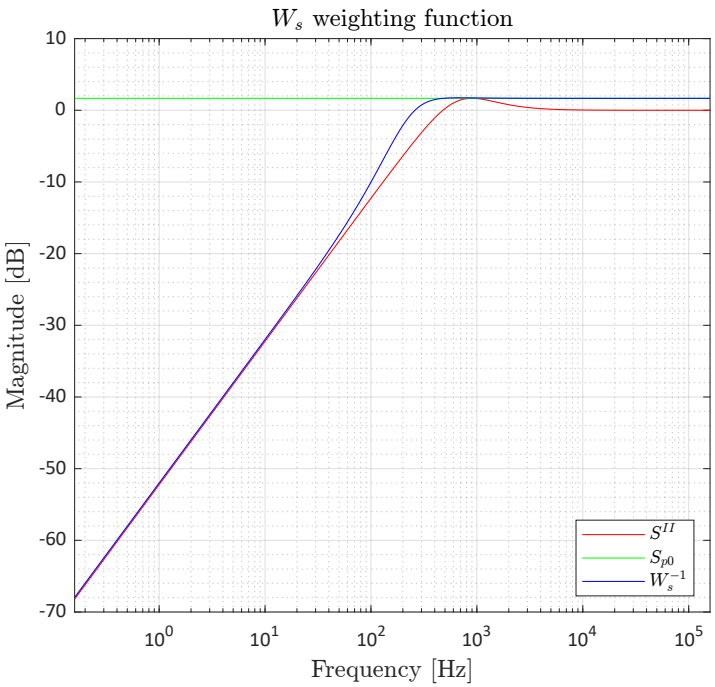


Figure 4.5: Weighting function W_s^{-1}, with $a = 0.0004$, $\omega_1 = 800\,\text{rad/s}$ and $\omega_2 = 1556.7\,\text{rad/s}$

$$W_s^{-1}(s) = \frac{1.2117s(s+800)}{s^2 + 2201s + 2.423e06} \tag{4.9}$$

$$S^{II}(s) = \frac{s(s+7013)}{s^2 + 7013s + 1.802e07} \tag{4.10}$$

About W_t^{-1} the discussion is the same: as for W_s^{-1}, W_t^{-1} is constructed using Butterworth polynomials.

$$W_t^{-1}(s) = \frac{K_t}{1 + 1.414s/\omega_t + (s/\omega_t)^2}$$

In the definition of W_t^{-1} there is only one constraint to be respected, that is T_{p0}, and so K_t must be equal to T_{p0}. However, two complex conjugate poles are included at a frequency of 15 kHz, that is 1/10 of the switching frequency: this is important in order to avoid too high crossover frequencies. The final weighting function is shown in equation (4.11), with a graphical representation provided in figure 4.6.

$$W_t^{-1}(s) = \frac{9.5402e09}{s^2 + 1.333e05s + 8.883e09} \tag{4.11}$$

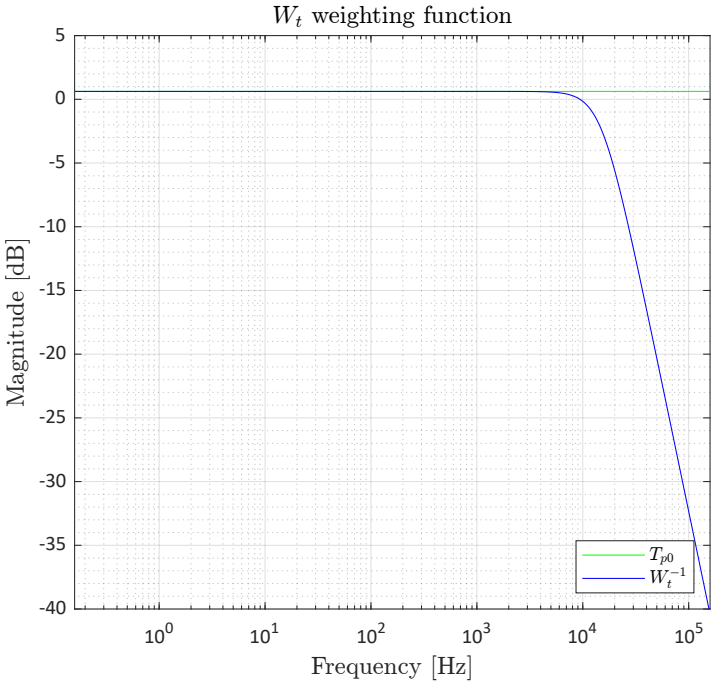


Figure 4.6: Weighting function W_t^{-1}, with $K_t = 1.0740$ and $\omega_t = 94248\,\text{rad/s}$

Then, also W_u is exploited: W_u has to take into account not only the dynamic uncertainty introduced with the choice of the nominal plant (the nominal plant was simplified in order to have a strictly proper transfer function) but also the variation of the working point, that is specifically related to the NIBB converter. In order to build the function W_u, a multiplicative uncertainty model set M_m is chosen, as described in (4.12),

$$M_m = \{G_{p,i}(s) : G_{p,i}(s) = G_{pn}(s)(1 + W_u(s)\Delta(s)), \|\Delta(s)\|_\infty \leq 1\} \tag{4.12}$$

where $\Delta(s)$ is any suitable transfer function with infinity norm less than or equal to 1 and $G_{p,i}(s)$ is a generic transfer function of the set that expresses the behavior of the plant. Developing the relationship expressed by (4.12), W_u can be defined with the inequality in (4.13).

$$|W_u(s)| \geq \left| \frac{G_{p,i}(s) - G_{pn}(s)}{G_{pn}(s)} \right| \tag{4.13}$$

Using for $G_{p,i}(s)$ the same transfer functions considered in figure 4.3, W_u is constructed in order to be above the uncertainty set, as shown in figure 4.7.

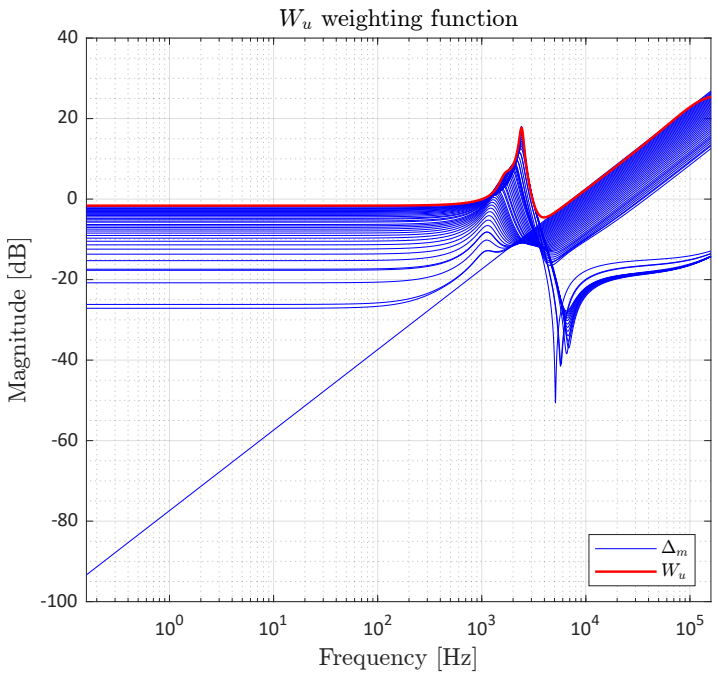


Figure 4.7: Weighting function $W_u(s)$ and $\Delta(s)$ set for CT design

$$
\begin{aligned}
W_u(s) &= W_{u1}(s) \cdot W_{u2}(s) \\
W_{u1}(s) &= \frac{11.157(s+7.104e06)(s+9.588e05)(s+2.156e04)}{(s+4.558e06)(s^2+3079s+1.079e08)} \\
W_{u2}(s) &= \frac{(s^2+4514s+1.054e08)(s^2+1.156e04s+4.449e08)}{(s^2+1394s+2.313e08)(s^2+1.262e06s+8.147e11)}
\end{aligned}
\tag{4.14}
$$

Now it is possible to define the weighting functions that will be used for the design. As written before, W_1 and W_2 can be chosen as equal to the previous weighting functions introduced. However, they are only a starting point and they can be changed in order to obtain a controller that accomplishes the desired requirements in the time domain. The transfer functions chosen are represented in figure 4.8 and shown in equations (4.15) and (4.16).

$$W_1(s) = \frac{0.82528(s^2+2201s+2.423e06)}{s(s+800)} \tag{4.15}$$

$$W_2(s) = 7.7589e-08(s+6000)^2 \tag{4.16}$$

However, these two functions cannot be used as they are in the design. As a matter of facts, W_1 is not stable and it is necessary to eliminate the pole in zero.

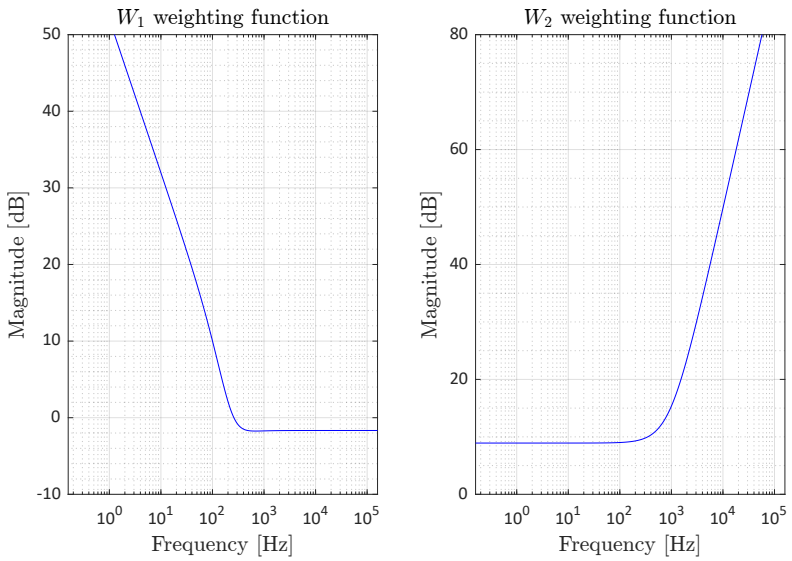


Figure 4.8: Weighting function W_1 and W_2

In order to do this, the pole in zero is substituted with a pole at a small frequency, that is $\lambda = 0.01\omega_c$. Moreover, W_2 is not proper and this could be a problem with the command `linmod`, that is used in order to derive the state space matrices of the generalized plant M. So, in M a modified W_2 with no zeros is used and then the two zeros are appended with the command `sderiv` to the system matrix representation of M, obtained using `ltisys`.

$$W_{1,mod}(s) = \frac{0.82528(s^2 + 2201s + 2.423e06)}{(s + 800)(s + 24.28)} \tag{4.17}$$

$$W_{2,mod}(s) = 2.7932 \tag{4.18}$$

The design is completed using the command `hinflmi` and providing the state space representation and then the transfer function of the controller.

Listing 4.1: Code for design procedure

```
[Am,Bm,Cm,Dm] = linmod('Generalized_Plant');
M = ltisys(Am,Bm,Cm,Dm);
M = sderiv(M,2,[1/pT 1]);
M = sderiv(M,2,[1/pT 1]);
[gopt,Cmod] = hinflmi(M,[1 1],0,1e-2,[0 0 1e-3 1]);
[Ac,Bc,Cc,Dc] = ltiss(Cmod);
Cmod = ss(Ac,Bc,Cc,Dc);
Cmod = zpk(Cmod);
```

The transfer function of the computed controller is:

$$C_{mod}(s) = \frac{2.1176e05(s + 893.5)(s^2 + 3376s + 4.563e07))}{(s + 7.818e06)(s + 1.223e04)(s + 800)(s + 24.28)}$$

C_{mod} has to be "cleaned", removing high frequency poles, poles and zeros with close values and adding the pole in zero needed for the zero tracking error requirement. The final controller is shown in (4.19), followed by the Nichols diagram of the nominal open loop transfer function (see figure 4.9).

$$C(s) = \frac{1.9058e05(s^2 + 3376s + 4.563e07)}{s(s + 7.818e06)(s + 1.223e04)} \tag{4.19}$$

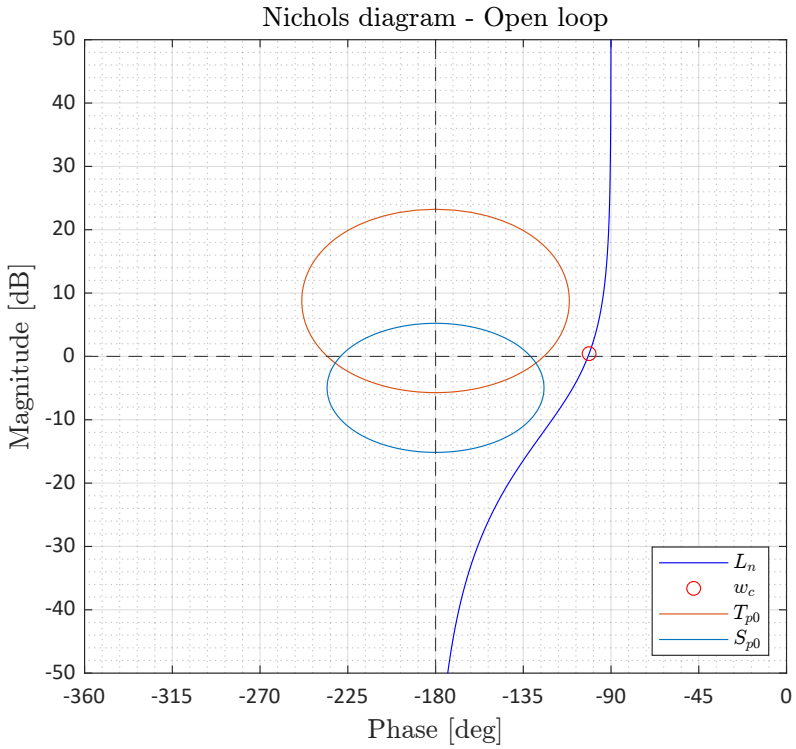


Figure 4.9: Nichols diagram of the nominal open loop transfer function $L_n(s)$

In figure 4.9, is possible to observe that the open loop function does not intersect the loci of T_{p0} and S_{p0}, that is important in order to ensure the achievement of the requirements expressed by T_{p0} and S_{p0}. Also, the crossover frequency is equal to the one desired, guaranteeing the requested performances.

4.3.1 Results

An important check, that can be done before the simulation, is the observation of the conditions related to robust stability and nominal performance (equation (4.3)). This can be easily done through graphical representation.

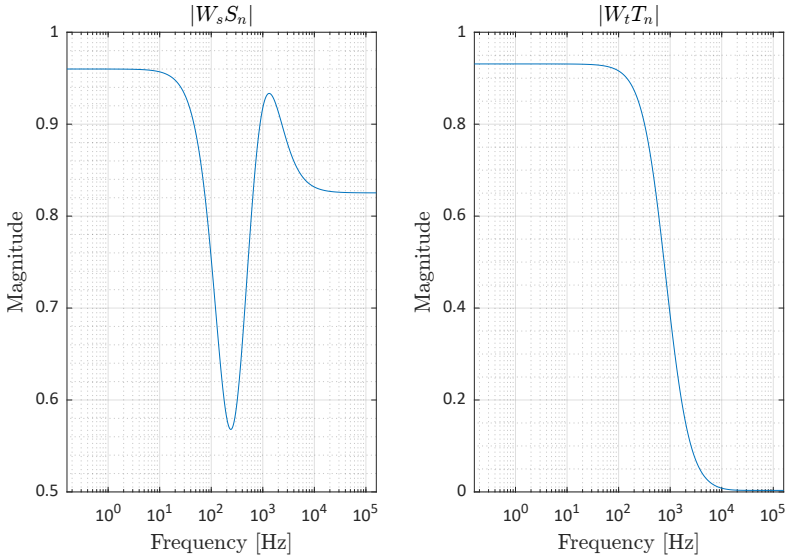


Figure 4.10: Diagrams of nominal performance conditions for CT design

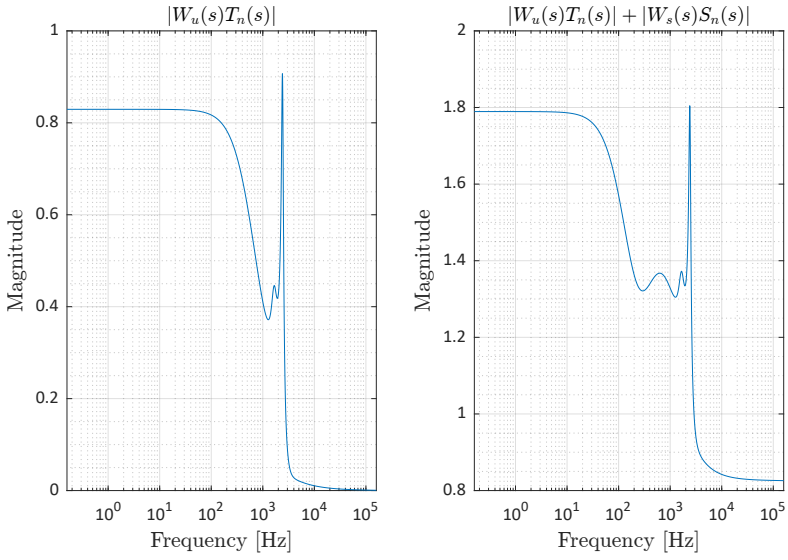


Figure 4.11: Diagrams of robust stability and robust performance conditions for CT design

From figure 4.10 the information provided is that the requirements in time are fully respected applying the designed controller to the nominal transfer function. In figure 4.11, the plot related to the robust stability condition is placed side by side to another plot related to the "robust performance". The robust performance condition expresses the capability of the controller to achieve the required performance also when applied to the real uncertain plant. Indeed, for a multiplicative uncertainty set, this condition is accomplished if the inequality of equation (4.20) is true.

$$\| |W_u(s)T_n(s)| + |W_s(s)S_n(s)| \|_\infty < 1 \tag{4.20}$$

As shown from the figure, the robust stability is respected, ensuring the stability of the controller for each working point in the range $v_{in} \in [8\ 60]\,\mathrm{V}$. Instead, the robust performance condition is not respected: this was quite expected, thus because is quite difficult to ensure the same performances of the system in buck mode for the system in boost mode, much slower than the first one.

In addition, it is also possible to verify the robustness of the single transfer function that was modified in order to obtain the nominal function used for the design procedure. In this case, the weighting function $W_{u,n}$ is obtained using only the nominal transfer function G_{pn} and the initial plant transfer function G_p, as shown in equation (4.21).

$$W_{u,n}(s) = \frac{G_p(s) - G_{pn}(s)}{G_{pn}(s)} \tag{4.21}$$

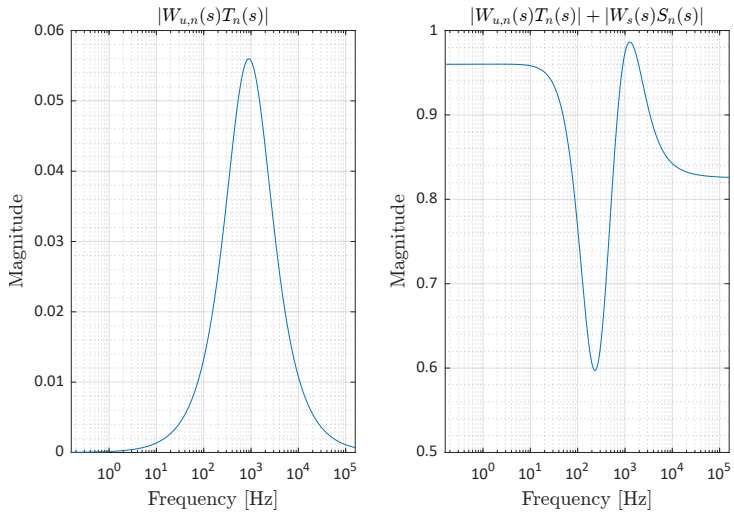


Figure 4.12: Diagrams of robust stability and robust performance conditions for the single function in CT design

As a matter of facts, modifying the plant transfer function, a dynamic uncertainty is introduced, that can be described with the function $W_{u,n}$. Figure 4.12 shows that, relatively to the single transfer function, the controller respects the robust performance and the robust stability conditions.

The phase and the gain margins are computed, providing and additional check for the robustness of the controller. In table 4.2 the same four points used in the previous chapter are considered (i.e. $v_{in} \in [8154060]$, with maximum power transfer condition).

W.P.	γ [dB]	ϕ [deg]	f_c [Hz]
A	∞	89.7	72.1
B	∞	89.6	72.3
C	∞	87.8	189.4
D	∞	81.3	405.8

Table 4.2: Margins and crossover frequencies related to the H_∞ controller and evaluated in four different working points

The analysis of the phase margins confirms that the controller is robustly stable. Finally, in order to observe the performances, four simulations are performed in the same working points exploited before.

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	0.20	5.39	7.36
B	0.01	4.93	5.10
C	0.17	1.61	1.00
D	0.25	0.69	0.38

Table 4.3: Values of overshoot, rise time and settling time in each working point of interest for H_∞ controller in CT

Figures 4.13 - 4.16 and table 4.3, outline that the required performances, imposed for the design, are not achieved in all the working points. However the H_∞ controller provides a significant improvement with respect to the PI controller described in chapter 3.

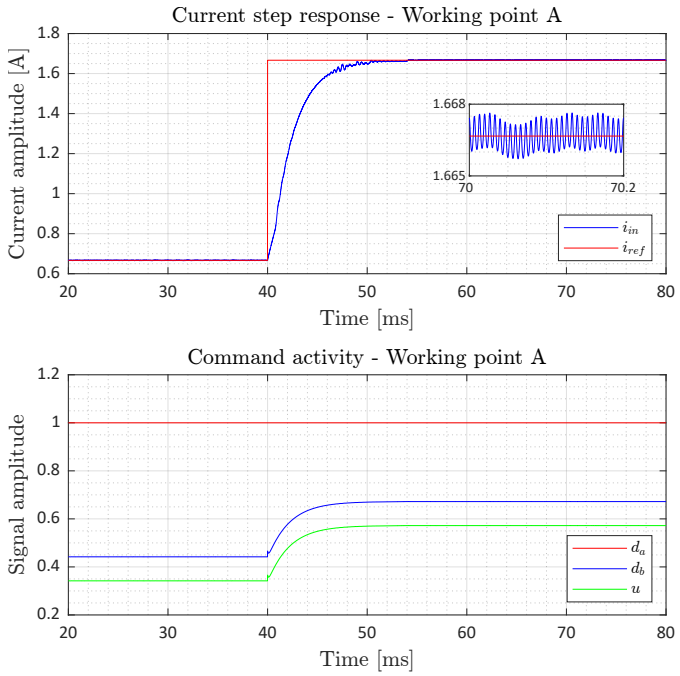


Figure 4.13: Simulation in working point A (boost) with H_∞ controller in CT

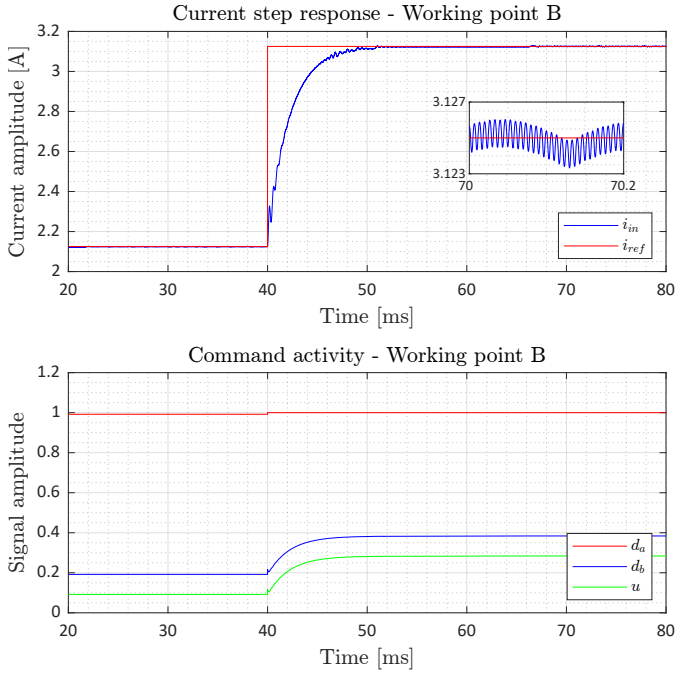


Figure 4.14: Simulation in working point B (boost) with H_∞ controller in CT

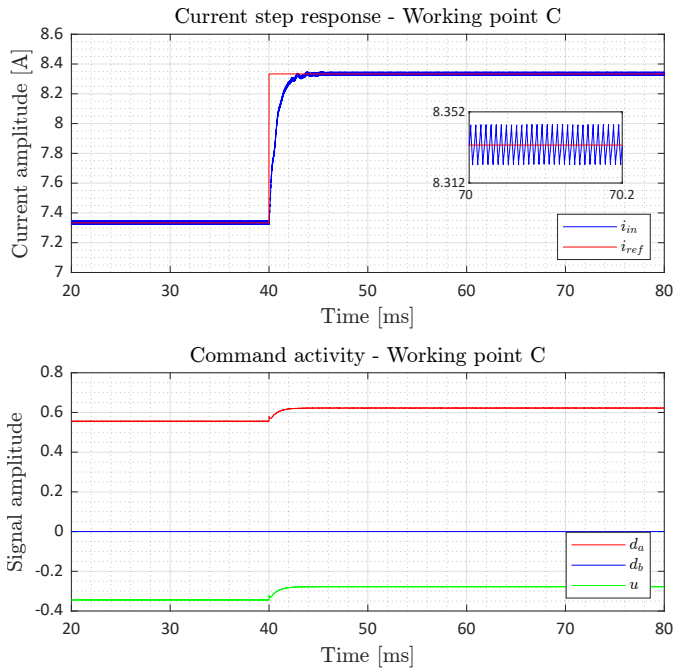


Figure 4.15: Simulation in working point C (buck) with H_∞ controller in CT

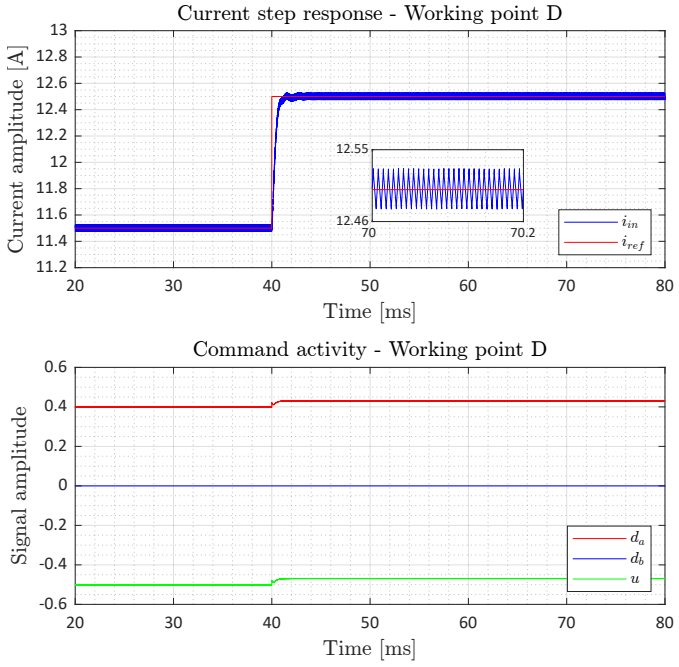


Figure 4.16: Simulation in working point D (buck) with H_∞ controller in CT

4.4 Discrete time design

The DT translation of the controller design is performed using the emulation technique. The idea behind the emulation is to design the controller in CT taking into account the dynamic of the A/D converter and of ZOH filter as part of the plant.

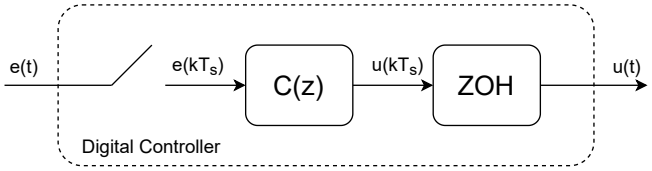


Figure 4.17: Digital controller general schematic

The A/D converter and the ZOH filter dynamics are expressed with suitable transfer functions, shown in 4.22.

$$G_{A/D} = \frac{1}{T_s} \qquad G_{ZOH}(s) = \frac{1 - e^{-T_s s}}{s} \approx \frac{T_s}{1 + sT_s/2} \tag{4.22}$$

Then the obtained controller is discretized, using the Tustin discretization method: a bilinear transformation that allows to pass from the Laplace domain to the z-domain, as outlined in equation (4.23).

$$s = \frac{2}{T_s} \cdot \frac{1 - z^{-1}}{1 + z^{-1}} \tag{4.23}$$

The introduction of the ZOH filter and of the A/D converter defines a variation in the initial set of transfer functions considered. The new transfer functions of the plant are obtained as expressed in (4.24).

$$G'_{p,i}(s) = G_{p,i}(s) \cdot G_{A/D} \cdot G_{ZOH}(s) = G_{p,i}(s) \cdot \frac{1}{1 + sT_s/2} \tag{4.24}$$

So the new set of transfer functions can be build easily adding a pole at $\omega = 2/Ts$. This set is represented in figure 4.18. In order to be coherent with the CT design, the same working point with $v_{in} = 60\,\mathrm{V}$ is used for building the starting plant transfer function G'_p.

$$G'_p(s) = \frac{295.15(s + 5.721e06)(s + 9.572e04)(s + 4.738e04)}{(s + 9.767e04)(s + 6e04)(s^2 + 3376s + 4.564e07)}$$

Simplifying G'_p in order to have a suitable nominal function for the design, the same G_{pn} used for the CT design is obtained. A graphical comparison between G'_p and G_{pn} is provided in figure 4.19.

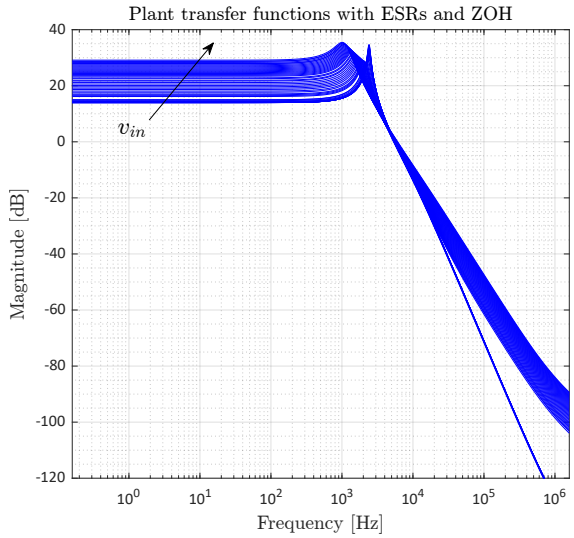


Figure 4.18: Transfer functions of 53 working points in the range $v_{in} \in [8\ 60]\,\mathrm{V}$ with ZOH filter and A/D converter

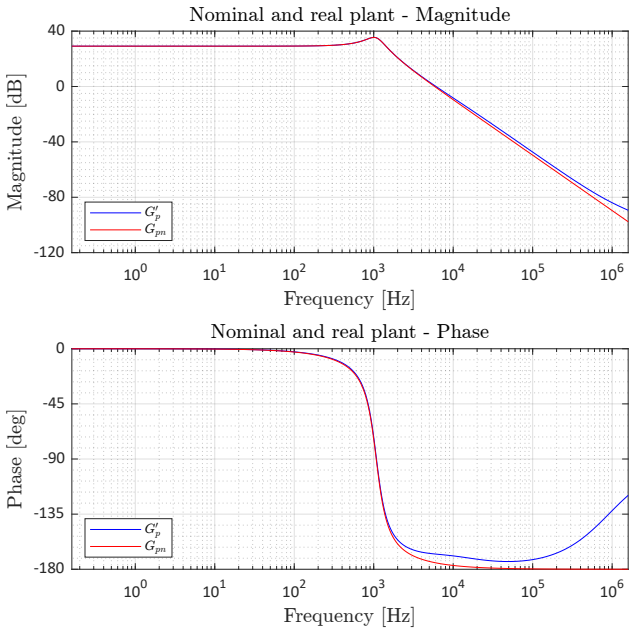


Figure 4.19: Graphical comparison of $G_{pn}(s)$ and $G_p(s)$ with ZOH filter and A/D converter

Because G_{pn} does not change, the design can be carried out in the same way of the CT case. W_s and W_t are built as shown in (4.9) and (4.11), while W_u is slightly different: even if G_{pn} is unchanged, the various $G'_{p,i}$ functions change at high frequency, due to the additional pole introduced. The uncertainty set is, as for the CT case, a multiplicative uncertainty set. The W_u derived is plotted in figure 4.20 and shown in equation (4.25).

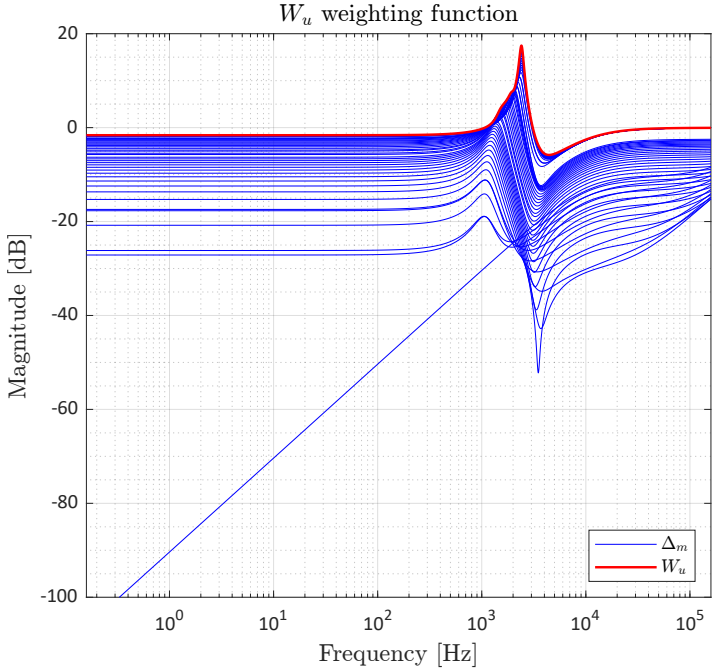


Figure 4.20: Weighting function $W_u(s)$ and $\Delta(s)$ set for discrete time design

$$
\begin{aligned}
W_u(s) &= W_{u1}(s) \cdot W_{u2}(s) \\
W_{u1}(s) &= \frac{0.99593(s + 2.38e04)(s^2 + 3143s + 8.536e07)}{(s + 5.951e04)(s^2 + 2706s + 8.925e07)} \\
W_{u2}(s) &= \frac{(s^2 + 1953s + 1.568e08)(s^2 + 1.482e04s + 4.86e08)}{(s^2 + 1874s + 1.524e08)(s^2 + 1370s + 2.296e08)}
\end{aligned}
\tag{4.25}
$$

Even if W_u is different, the suitable W_1 and W_2 do not change with respect to the CT design case. In the end, the controller obtained is equal to the one obtained with the previous CT design (see (4.19)). The digital controller, discretized with the Tustin method is shown in equation (4.26).

$$
C(z) = \frac{0.021479(z + 1)(z^2 - 1.847z + 0.8947)}{(z - 0.6613)(z - 1)(z + 0.9848)} \tag{4.26}
$$

However, the discrete transfer function obtained for the controller is not optimal. As a matter of facts, there are a negative zero and a negative pole that introduce alternatives modes, able to produce ringing effects (i.e. the control variable presents a series oscillations both in the transient and the steady state phase). In order to avoid the ringing, the zero and the pole are canceled, thanks also to their numerical values, very close to each other.

$$C(z) = \frac{0.021479(z^2 - 1.847z + 0.8947)}{(z-1)(z-0.6613)} \tag{4.27}$$

Equation (4.27) shows the zero-pole-gain form of the final controller transfer function. In this case, the sampling time chosen is the same used for the discrete PI in chapter 3 (i.e. $T_s = 1/f_s$ with $f_s = 30\,\text{kHz}$).

4.4.1 Results

Because the controller obtained with emulation does not change with respect to the one designed for the CT case, the information about the nominal performance do not change. In this case, only the check on robustness and time performance are needed. The robust stability and the robust performance are evaluated through graphical analysis (see figure 4.21).

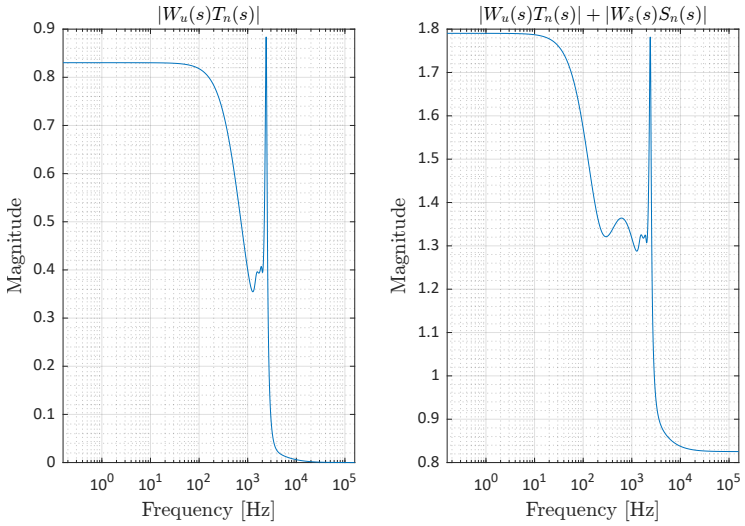


Figure 4.21: Diagrams of robust stability and robust performance conditions for DT design

There is no difference in terms of norms, thus because the pole added to the plant with the emulation technique is at a very high frequency, defining a shape change of W_u only at high frequency. The same discussion is true if the single plant

transfer function is considered, as shown in figure 4.22. Also the design for the DT case provides a controller that is robustly stable for all the working point analysed.

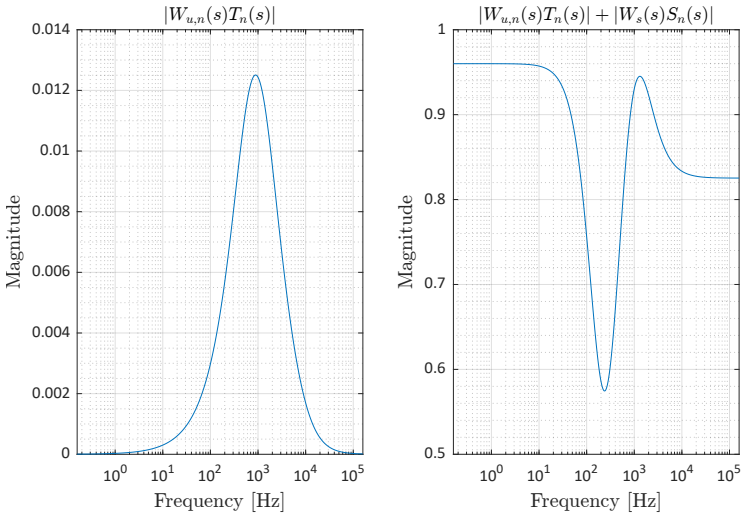


Figure 4.22: Diagrams of robust stability and robust performance conditions for the single function in DT design

Then a simulation for each working point is performed, in order to check the performances of the closed loop system. The results are shown in figures 4.23 - 4.26, with all the numerical values resumed in table 4.4.

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	0.17	5.40	6.98
B	0.09	4.93	5.12
C	0.21	1.55	0.98
D	0.31	0.68	0.38

Table 4.4: Values of overshoot, rise time and settling time in each working point of interest for H_∞ controller in DT

The results outline that, also for the DT case, the performances are significantly enhanced with respect to the simple PI controller.

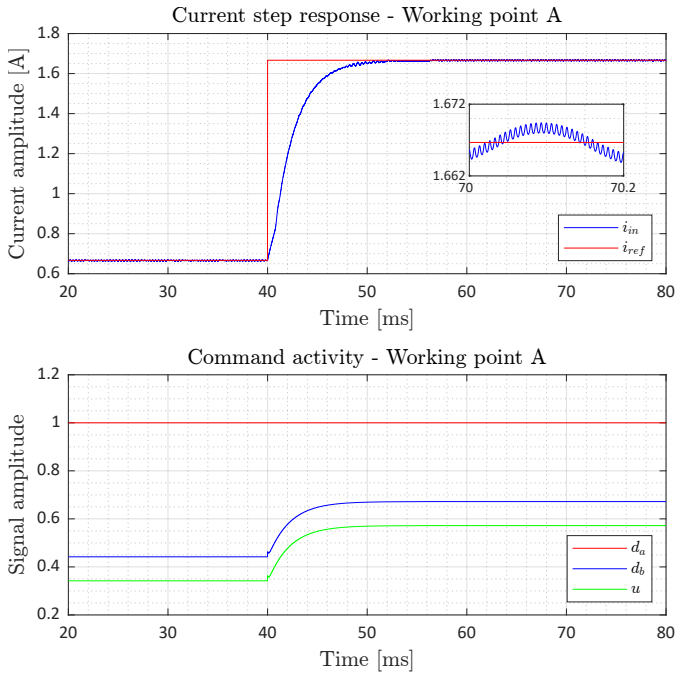


Figure 4.23: Simulation in working point A (boost) with H_∞ controller in DT

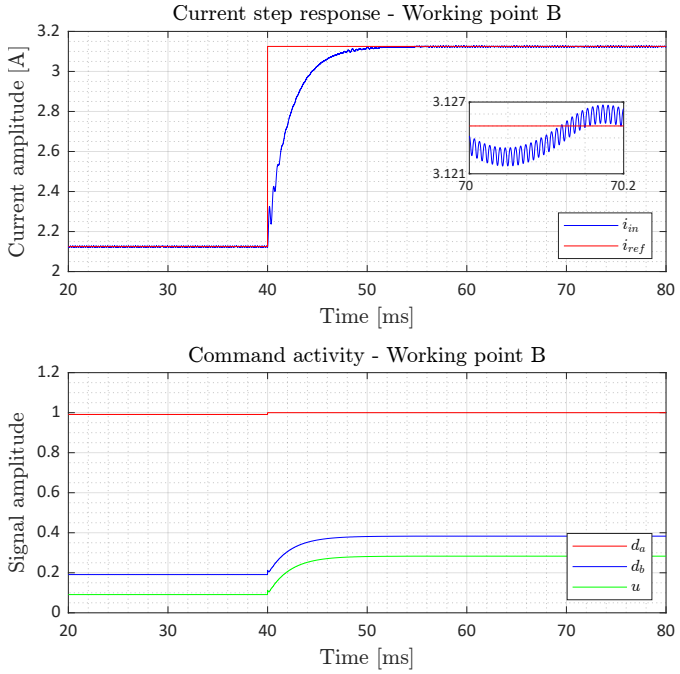


Figure 4.24: Simulation in working point B (boost) with H_∞ controller in DT

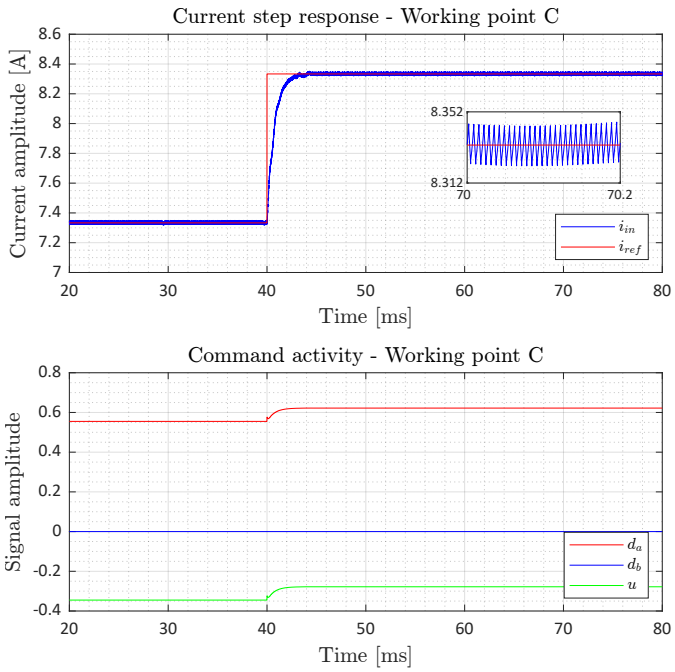


Figure 4.25: Simulation in working point C (buck) with H_∞ controller in DT

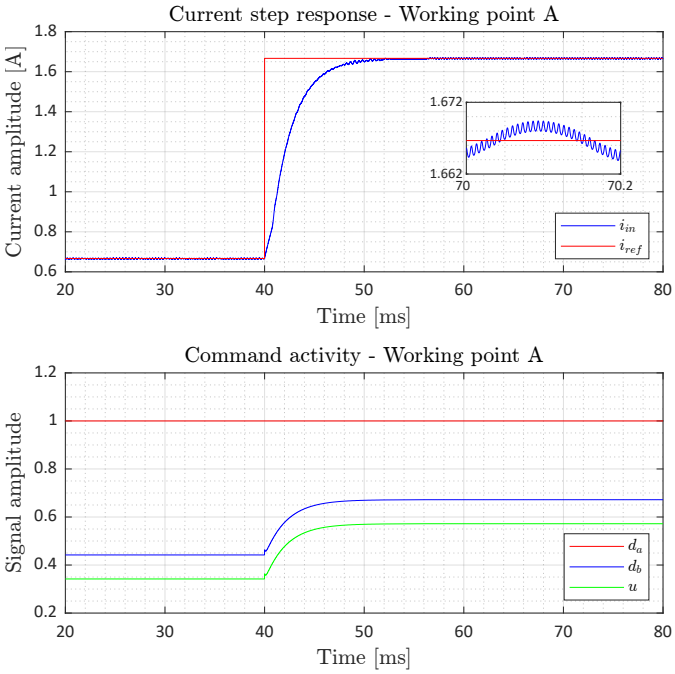


Figure 4.26: Simulation in working point D (buck) with H_∞ controller in DT

Chapter 5

Linear quadratic regulator control design

All the control techniques described until now consist of output-feedback controllers, that exploit the input current i_{in} of the converter as control variable. The PI and the H_∞ design techniques provide very interesting performances with large phase margins, that indicate good robustness property of the closed-loop system. However, there exist control laws that are able to use more than one variable for the definition of the control action, providing high performances and good margins. One example is the Linear Quadratic Regulator (**LQR**), a control technique that exploits the information provided by the states of the system in order to define a state-feedback controller.

Here, after discussing the state of the art, the design of an LQR controller is provided, both in continuous and discrete time, paying attention to the theoretical fundamentals and providing suitable simulations for time performance analysis.

5.1 State of art

The LQR technique is not the only control law able to establish a state-feedback control. A very similar construction can be provided by means of a simple pole placement. Both the pole placement and the LQR place the poles of the closed loop system in a desired frequency, using the state as control variable: the main difference is in the procedure that allows to choose the positions of the poles. With the LQR, the poles are placed solving an optimization problem, while with the pole placement the poles values are decided directly by the designer.

The pole placement technique is exploited in [27]: in this work, after deriving the model and the transfer function of a buck-boost converter with the flow graph technique and the Mason gain formula, the pole placement design is applied in order to derive a suitable control law. The proposed controller defines the poles

location of the closed loop system, aiming to achieve a prototype time domain response (e.g. filter pole locations). Instead, the LQR controller is based on the solution of an optimization problem, using a cost function that is able to take into account for both the state and the command input optimization. [28] provides an accurate description of LQR application for switching DC-DC converters. [29] and [30] develop the design of an LQR controller for a buck converter, focusing on the concepts of robustness and stability.

In [29] the authors outline how an LQR controller provides many advantages in the field of power conversion, like a phase margin larger than 60 deg, that is a general standard requirement in most of power electronics applications. However, the LQR controller cannot ensure robust stability in the case of high uncertain systems. So, in order to improve the robustness property of the LQR, the authors develop the problem through LMIs and numerically solve it by means of convex optimization algorithms (i.e. interior point algorithm). The LMI formulation has a lot of advantages, because it allows to take into account for multiple plants (robustness) and to include different design requirements.

[30] defines an LQR design procedure that aims to ensure rejection of disturbances on the input voltage, a well defined settling time and robustness to variations of the load in a well known interval. In order to obtain the disturbances rejection, the authors exploit an analysis based on the H_∞ norm of the closed loop system, computed by means of LMIs formulation. The requirement on the settling time is achieved imposing a bound on the real part of the eigenvalues of the closed loop system. Finally, the robustness with respect to the load variations is guaranteed through the definition of a suitable Lyapunov function computed with LMIs.

[31], instead, describes a new kind of advanced non inverting buck boost converter, composed by a boost converter and a buck converter linked through a magnetic coupling (i.e. this allows to remove right half-plane zeros in the transfer function of the plant, increasing the bandwidth and improving the efficiency). Then the classical LQR technique with augmented state is introduced in order to design a suitable and robust controller.

5.2 Introduction

The LQR, also known as Linear Quadratic optimal control, is a control technique that takes into account for the trade-off between state performances and command activity. This is important in many applications where control requirements cannot be totally described in terms of poles location.

As a matter of facts, the main feature of the LQR is that the location of the poles is "optimally chosen", resolving an optimization problem, in which it is possible to take into account also for energy and amplitude limitations. This can be achieved using a suitable cost function that provides information about both the state and the command activity of the system itself.

5.2.1 Continuous time case

Considering a general dynamical system, described by the following state space representation in CT:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{5.1}$$

the LQR problem can be formulated as the research of a suitable command input u^* able to minimize the chosen cost function J.

$$\begin{aligned} u^*(t) &= \arg \min_{u(t), t\in[0,\infty)} J(u) \\ \text{s.t. } \dot{x}(t) &= Ax(t) + Bu(t) \end{aligned}$$

The cost function is written in equation (5.2) and it is function of the command input u (i.e. the state x is a function of the command itself).

$$J(u) = \int_{\tau=0}^{\infty} (x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau)) \cdot d\tau \tag{5.2}$$

Observing J, it is possible to provide a physical meaning for the cost function itself: it is the summation of the squares of the weighted two norms of x and u, that can be seen as the summation of the energies of the states and of the command activity.

$$J(u) = \|x\|_{Q,2}^2 + \|u\|_{R,2}^2$$

The control law is obtained minimizing the "weighted energies" of u and x, subject to the constraint that u and x must satisfy the dynamic equations. Moreover, minimizing the norm of x ensures closed-loop stability, because if x is minimized it cannot diverge. Q and R are the design parameters, set as diagonal matrices and chosen accordingly to the desired trade-off: increasing the weight of one component of the cost function means to require a greater minimization of that component itself in the cost function.

$$Q = Q^T \geq 0 \quad \text{and} \quad R = R^T > 0$$

Given the LTI system described by the state space representation (5.1), if the system is controllable (i.e. the controllability matrix is full rank) then the optimal solution u^* is given by:

$$\begin{aligned} u^*(t) &= -R^{-1}B^T P \cdot x(t) \\ &= -K \cdot x(t) \end{aligned}$$

where $P = P^T > 0$ is the solution of the Continuous Algebraic Riccati Equation (**CARE**) shown in (5.3). The control law defines a state feedback control architecture, as depicted in figure 5.1.

$$Q - PBR^{-1}B^T P^T + PA + A^T P = 0 \tag{5.3}$$

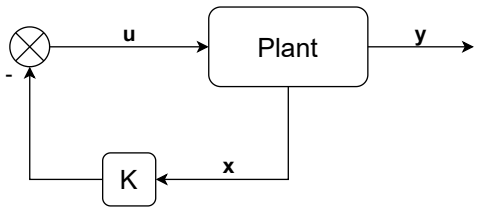


Figure 5.1: General block diagram of the LQR state feedback

Moreover, a further condition for the stability property of this controller can be provided. Expressing Q as $Q = C_q^T C_q$ where C_q is the Cholesky factor of Q, if the couple (A, B) is controllable and the couple (A, C_q) is observable (i.e. the observability matrix is full rank) then the closed loop system described by the state equation $\dot{x}(t) = (A - BK)x(t)$ is asymptotically stable.

5.2.2 Discrete time case

The formulation of the LQR problem in DT is similar to the formulation provided for the continuous time case. Taking into account the DT dynamical system written in (5.4) for a generic time instant k,

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned} \tag{5.4}$$

the optimization problem, also defined as infinite horizon discrete time linear quadratic optimal control problem, is reformulated as follows:

$$u^*(k) = \arg \min_{u(k), k \in [0,\infty)} J(x(k), U(k))$$

$$\text{s.t. } x(k+1) = Ax(k) + Bu(k)$$

with the cost function J translated in DT as :

$$J(x(k), U(k)) = \sum_{i=0}^{\infty} x^T(k+i)Qx(k+i) + u^T(k+i)Ru(k+i) \tag{5.5}$$

In this case, as written in (5.5),the cost function depends on x(k) and U(k), with U(k) defined as the command input sequence.

$$\begin{aligned} &U(k) = [u(k)\ \ u(k+1)\ \ \ldots\ \ u(k+H_p-1)] \\ &\text{with } H_p \to \infty \end{aligned}$$

Q and R are the same design parameters introduced before and the solution to this optimization problem can be found in a similar way to the CT case, introducing the discrete equivalent of the CARE. Given the LTI system described by the state space representation (5.4), if the system is controllable then the optimal solution u^* is given by:

$$\begin{aligned} u^*(k) &= -(R + B^T P B)^{-1} B^T P A \cdot x(k) \\ &= -K \cdot x(k) \end{aligned}$$

where $P = P^T > 0$ is the solution of the Discrete Algebraic Riccati Equation (**DARE**) written in (5.6). The control law defines a state-feedback control architecture, as depicted in figure 5.1.

$$P = A^T P A + Q - A^T P B (R + B^T P B)^{-1} B^T P A \tag{5.6}$$

Also, the same stability condition exploited for the LQR in CT can be provided for the DT case: if the couple (A, B) is controllable and the couple (A, C_q) is observable then the closed loop system described by the state equation $x(k+1) = (A - BK)x(k)$ is asymptotically stable. A detailed description of the theory related to the linear optimal control is provided in [32].

5.3 Continuous time design

The design of an LQR controller in CT starts from the definition of a suitable state space representation. In this case, the model without ESRs is taken into account:

$$\begin{aligned} \dot{x}(t) &= f^{CT}(x(t), d(t), v(t)) \\ y(t) &= h^{CT}(x(t), v(t)) \end{aligned} \tag{5.7}$$

In (5.7), $x(t) = [v_{C_{in}}(t)\ v_{C_{out}}(t)\ i_L(t)]$ is the state vector, $d(t) = [d_a(t)\ d_b(t)]$ is the command input vector and $v(t) = [v_{in}(t)\ v_{out}(t)]$ is the electrical input vector. The expressions of f^{CT} and h^{CT} are shown in equation (5.8).

$$f^{CT} = \begin{bmatrix} \frac{v_{in}(t)-v_{C_{in}}(t)}{R_{in}C_{in}} - \frac{d_a(t)i_L(t)}{C_{in}} \\ \frac{v_{out}(t)-v_{C_{out}}(t)}{R_{out}C_{out}} - \frac{(1-d_b(t))i_L(t)}{C_{out}} \\ \frac{d_a(t)v_{C_{in}}(t)}{L} - \frac{(1-d_b(t))v_{C_{out}}(t)}{L} \end{bmatrix} \quad h^{CT} = \frac{v_{in}(t) - v_{C_{in}}(t)}{R_{in}} \tag{5.8}$$

The choice of this simplified model is related to the linearization process that is needed for the LQR control application (i.e. the LQR is a linear control technique). Using the state space model without ESR effects (its accuracy was verified in chapter 2), the linearization appears simpler to be used and represented. Because the model changes depending on the working condition of the system, two controllers are designed: one for the buck case and one for the boost case.

5.3.1 Buck case

With the linearization, the objective is to define a suitable linear system, as in (5.1). For this reason, d_a and d_b are changed taking into account the mathematical relationships introduced by the PWM (see equations (2.23) and (2.24)). In the buck case, d_a and d_b are set as:

$$\begin{aligned} d_a &= u \cdot K1 + K2 \\ d_b &= 0 \end{aligned} \tag{5.9}$$

Then the linearization procedure is performed, considering $v_{in} = 60\,\mathrm{V}$ and maximum power transfer condition. The computed matrices are written in equation (5.10), where D_a, $V_{C_{in}}$ and I_L are the big signals defined by the working point.

$$\begin{aligned} A_\ell &= \begin{bmatrix} -\frac{1}{R_{in}C_{in}} & 0 & -\frac{D_a}{C_{in}} \\ 0 & -\frac{1}{R_{out}C_{out}} & \frac{1}{C_{out}} \\ \frac{D_a}{L} & -\frac{1}{L} & 0 \end{bmatrix} \quad B_\ell = \begin{bmatrix} -\frac{I_L K_1}{C_{in}} \\ 0 \\ \frac{V_{C_{in}} K_1}{L} \end{bmatrix} \\ C_\ell &= \begin{bmatrix} -\frac{1}{R_{in}} & 0 & 0 \end{bmatrix} \end{aligned} \tag{5.10}$$

The design of the controller needs the verification of the controllability: in Matlab, the controllability matrix can be build with the command `Mc = ctrb(A,B)` and the rank can be computed with the command `rank(Mc)`. Because, with the matrices introduced in (5.10), the system is completely controllable, the design can be performed.

An integrative contribute has to be introduced, for guaranteeing a zero tracking error. In this case, a state q (i.e. the integral of the tracking error) is added to the system, that is said to be "augmented". In order to define the gain to be applied to the state q, it is necessary to augment the state space representation: in this way, the LQR optimization problem will take into account also the additional state. The augmented system is described by the matrices A_a and B_a written in (5.11).

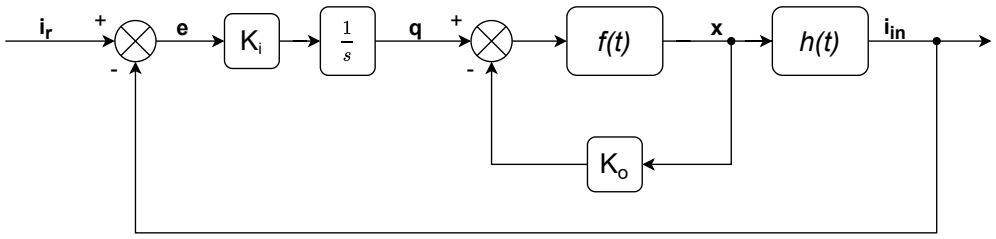


Figure 5.2: Block diagram of the augmented system with LQR control in CT

$$A_a = \begin{bmatrix} A_\ell & 0_{3x1} \\ C_\ell & 0 \end{bmatrix} \quad B_a = \begin{bmatrix} B_\ell \\ 0 \end{bmatrix} \tag{5.11}$$

Finally, the tuning parameters are introduced. Q is the augmented tuning matrix related to the state, composed by Q_o and q_i, while R is a scalar referred to the single input u. Q_o is the diagonal matrix related to the state x, while q_i is a single entry related to the integrative state q. The values of Q and R are listed in table 5.1.

$$Q = \begin{bmatrix} Q_o & 0_{3x1} \\ 0_{1x3} & q_i \end{bmatrix} \quad Q_o = \begin{bmatrix} q_1 & 0 & 0 \\ 0 & q_2 & 0 \\ 0 & 0 & q_3 \end{bmatrix}$$

Weight	q_1	q_2	q_3	q_i	R
Value	1e2	1	1	1e11	1e6

Table 5.1: Weights values for LQR control design, in CT buck case

The controller gain is designed using the command `K = lqr(Aa,Ba,Q,R)`. The gain K is a row vector that contains the gain K_o related to the state x and the gain K_i related to the additional state q, as written in (5.12). The numerical result is shown in (5.13).

$$K = \begin{bmatrix} K_o & K_i \end{bmatrix} \tag{5.12}$$

$$K_{buck}^{CT} = \begin{bmatrix} -0.0201 & -0.0025 & 0.0028 & 316.2278 \end{bmatrix} \tag{5.13}$$

As described by [32], the LQR controller is characterized by a phase margin that is greater than 60 deg and so a check on the margins is not necessary. The only check needed is about the asymptotic stability: considering C_q as the Cholesky factor of Q_o, the observability matrix `Mo = obsv(A,Cq)` is full rank. So the system is asymptotically stable (this is true also for the augmented system).

5.3.2 Boost case

For the boost working mode, the design procedure is exactly the same: the only difference is on the definition of the linearized matrices A and B. Indeed, in this case the PWM relationships are different:

$$\begin{aligned} d_a &= 1 \\ d_b &= u \cdot K3 + K4 \end{aligned} \tag{5.14}$$

After the linearization procedure, the matrices A_ℓ, B_ℓ and C_ℓ are computed, as written in (5.15), and then the controllability is checked. Because the system is controllable, the design can start, exploiting again the additional integrative state q, in order to ensure zero tracking error with a step reference.

$$\begin{aligned} A_\ell &= \begin{bmatrix} -\frac{1}{R_{in}C_{in}} & 0 & -\frac{1}{C_{in}} \\ 0 & -\frac{1}{R_{out}C_{out}} & -\frac{D_b-1}{C_{out}} \\ \frac{1}{L} & \frac{D_b-1}{L} & 0 \end{bmatrix} \quad B_\ell = \begin{bmatrix} 0 \\ -\frac{I_L K_3}{C_{out}} \\ \frac{V_{C_{out}} K_3}{L} \end{bmatrix} \\ C_\ell &= \begin{bmatrix} -\frac{1}{R_{in}} & 0 & 0 \end{bmatrix} \end{aligned} \tag{5.15}$$

The working point chosen for the design is $v_{in} = 8\,\mathrm{V}$, with maximum power transfer condition. The matrix Q and the scalar R are set; their values are summarized in table 5.2.

Weight	q_1	q_2	q_3	q_i	R
Value	1e2	1	1	1e11	1e6

Table 5.2: Weights values for LQR control design, in CT boost case

The controller gain is obtained exploiting the `lqr` command. The numerical values of the controller are written in equation (5.16).

$$K_{boost}^{CT} = \begin{bmatrix} -0.0020 & -0.0009 & 0.0027 & 316.2278 \end{bmatrix} \tag{5.16}$$

Then the check on the asymptotic stability is performed: considering C_q as the Cholesky factor of Q_o, the observability matrix `Mo = obsv(A,Cq)` is full rank. So the system is asymptotically stable.

5.3.3 Results

In order to measure the performances of the two controllers, four simulations are performed (shown in figures 5.3 - 5.6). All the results related to rise time, settling time and percentage overshoot are collected in table 5.3. The tests for the points A and B are performed using the controller K_{boost}^{CT}, while the tests in points C and D are performed with the controller K_{buck}^{CT}.

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	0.61	1.58	2.06
B	0.47	1.42	1.46
C	0.18	0.79	0.39
D	0.63	0.40	0.31

Table 5.3: Values of overshoot, rise time and settling time in each working point of interest for LQR controller in CT

The results and the graphical representations show an interesting improvement on the performances: in all the working points tested, the rise time and the settling time are smaller with respect to the values obtained with PI and H_∞ controllers. However, the command is affected by large oscillations that could be problematic in the practical applications.

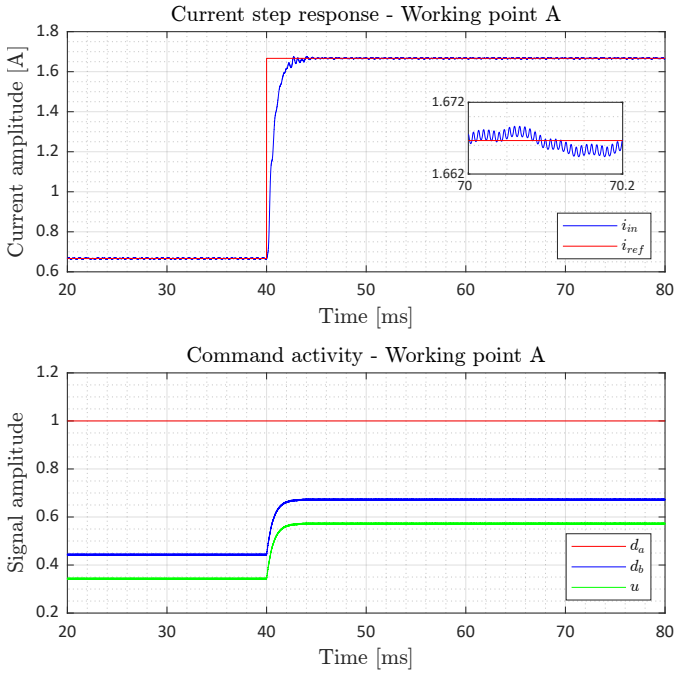


Figure 5.3: Simulation in working point A (boost) with LQR controller in CT

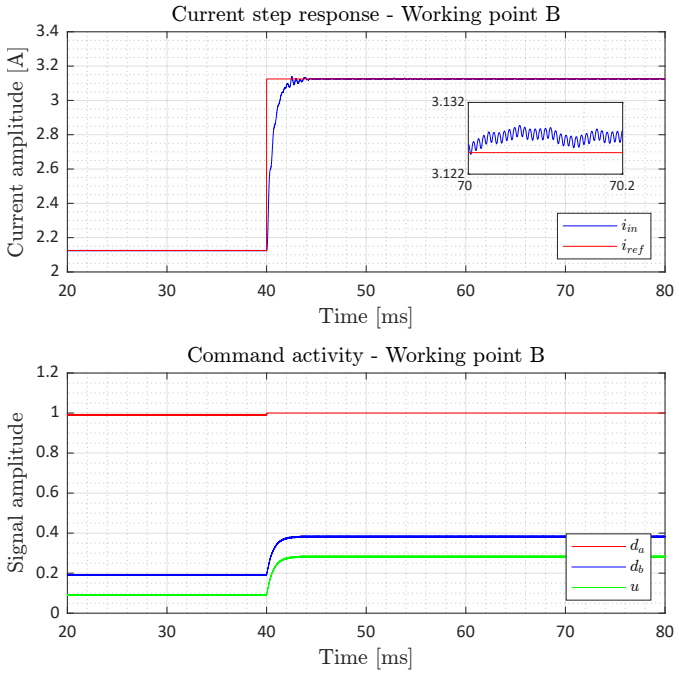


Figure 5.4: Simulation in working point B (boost) with LQR controller in CT

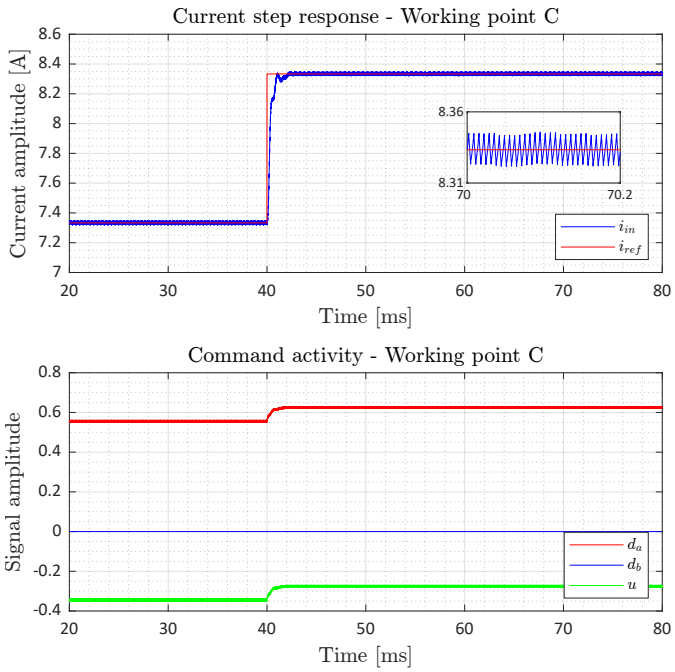


Figure 5.5: Simulation in working point C (buck) with LQR controller in CT

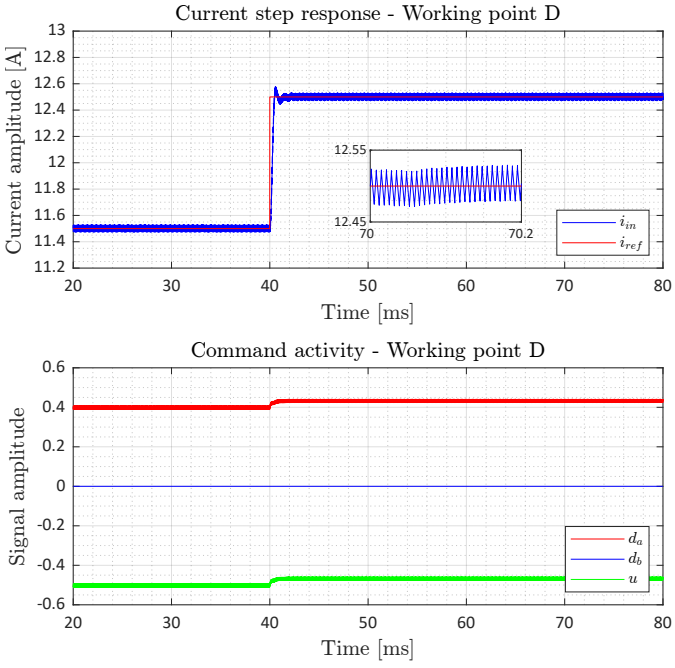


Figure 5.6: Simulation in working point D (buck) with LQR controller in CT

5.4 Discrete time design

The design of the LQR in DT is carried with a procedure similar to the one seen for the CT case. The main difference is related to the system description, that has to be translated in DT. The discretization can be developed starting from the state space representation expressed in chapter 2 (see (2.7)), that is reported also here for clarity sake. In this case, only the output $y = i_{in}$ is considered, while $v = [v_{in} \ v_{out}]$ is the input vector and $x = [v_{C_{in}} \ v_{C_{out}} \ i_L]$ is the state vector. The matrices values are resumed in (5.17) and (5.18).

$$\dot{x}(t) = Ax(t) + Bv(t)$$
$$y(t) = Cx(t) + Dv(t)$$

$$A = \begin{bmatrix} -\frac{1}{R_{in}C_{in}} & 0 & -\frac{d_a}{C_{in}} \\ 0 & -\frac{1}{R_{out}C_{out}} & -\frac{d_b-1}{C_{out}} \\ \frac{d_a}{L} & \frac{d_b-1}{L} & 0 \end{bmatrix} \quad B = \begin{bmatrix} \frac{1}{R_{in}C_{in}} & 0 \\ 0 & \frac{1}{R_{out}C_{out}} \\ 0 & 0 \end{bmatrix} \tag{5.17}$$

$$C = \begin{bmatrix} -\frac{1}{R_{in}} & 0 & 0 \end{bmatrix} \quad D = \begin{bmatrix} \frac{1}{R_{in}} & 0 \end{bmatrix} \tag{5.18}$$

The objective of the discretization is to derive a suitable state space representation in DT, described as follows:

$$x(k+1) = A_d x(k) + B_d v(k)$$
$$y(k) = C_d x(k) + D_d v(k)$$

The definition of C_d and D_d is simple, because they are exactly equal to C and D in CT. For A_d and B_d the computation procedure is different: they are derived applying the procedure of the Forward Euler Approximation. Considering T_s as the sampling time and I_{3x3} as the identity matrix, the discretized matrices are:

$$A_d = I_{3x3} + AT_s \quad B_d = BT_s$$

$$A_d = \begin{bmatrix} 1 - \frac{T_s}{R_{in}C_{in}} & 0 & -\frac{d_a T_s}{C_{in}} \\ 0 & 1 - \frac{T_s}{R_{out}C_{out}} & -\frac{(d_b-1)T_s}{C_{out}} \\ \frac{d_a T_s}{L} & \frac{(d_b-1)T_s}{L} & 1 \end{bmatrix} \quad B_d = \begin{bmatrix} \frac{T_s}{R_{in}C_{in}} & 0 \\ 0 & \frac{(d_b-1)T_s}{R_{out}C_{out}} \\ 0 & 0 \end{bmatrix} \tag{5.19}$$

Finally, the needed functions for the linearization procedure are derived, with f^{DT} and h^{DT} written as follows:

$$\begin{aligned} x(k+1) &= f^{DT}(x(k), d(k), v(k)) \\ y(k) &= h^{DT}(x(k), v(k)) \end{aligned} \tag{5.20}$$

$$f^{DT} = \begin{bmatrix} v_{C_{in}}(k)\left(1 - \frac{T_s}{R_{in}C_{in}}\right) - \frac{d_a(k) i_L(k) T_s}{C_{in}} + \frac{T_s v_{in}(k)}{R_{in}C_{in}} \\ v_{C_{out}}(k)\left(1 - \frac{T_s}{R_{out}C_{out}}\right) - \frac{(1-d_b(k)) i_L(k) T_s}{C_{out}} + \frac{T_s v_{out}(k)}{R_{out}C_{out}} \\ \frac{d_a(t) v_{C_{in}}(k) T_s}{L} - \frac{(1-d_b(t)) v_{C_{out}}(k) T_s}{L} + i_L(k) \end{bmatrix} \tag{5.21}$$

$$h^{DT} = \frac{v_{in}(k) - v_{C_{in}}(k)}{R_{in}}$$

5.4.1 Buck case

The linearization procedure does not change with respect to the CT case: for the buck, the working point considered is $v_{in} = 60\,\mathrm{V}$, with maximum power transfer. Using the state equations as written in (5.21) and the PWM relationship as written in (5.9), the linearized matrices are:

$$A_\ell = \begin{bmatrix} 1 - \frac{T_s}{R_{in}C_{in}} & 0 & -\frac{D_a T_s}{C_{in}} \\ 0 & 1 - \frac{T_s}{R_{out}C_{out}} & \frac{T_s}{C_{out}} \\ \frac{D_a T_s}{L} & -\frac{T_s}{L} & 1 \end{bmatrix} \quad B_\ell = \begin{bmatrix} -\frac{I_L K_1 T_s}{C_{in}} \\ 0 \\ \frac{V_{C_{in}} K_1 T_s}{L} \end{bmatrix} \tag{5.22}$$

$$C_\ell = \begin{bmatrix} -\frac{1}{R_{in}} & 0 & 0 \end{bmatrix}$$

As for the CT case, also in DT an integrative contribute is necessary in order to guarantee a zero tracking error. The design concept is the same: the additional state q is introduced and the system is augmented, however the matrices of the augmented system change, as shown in (5.23).

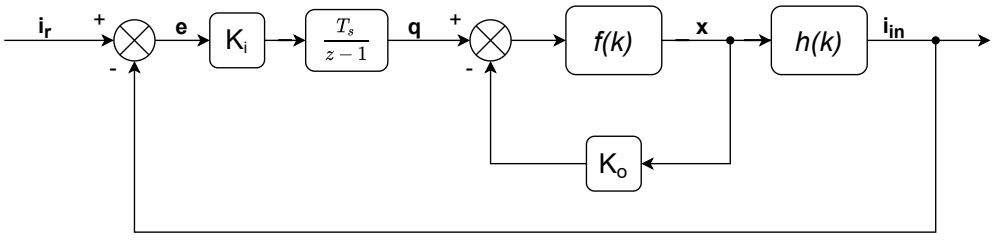


Figure 5.7: Block diagram of the augmented system with LQR control in DT

$$A_a = \begin{bmatrix} A_\ell & 0_{3x1} \\ C_\ell T_s & 1 \end{bmatrix} B_a = \begin{bmatrix} B_\ell \\ 0 \end{bmatrix} \tag{5.23}$$

Then, because the system is fully controllable, the design procedure can be performed, using the command `K = dlqr(Aa,Ba,Q,R)`. The construction of Q and R does not change. The weights used are listed in table 5.4 and the numerical values of the controller are written in equation (5.24).

Weight	q_1	q_2	q_3	q_i	R
Value	50	1	1	1e10	1e3

Table 5.4: Weights values for LQR control design, in DT buck case

$$K_{buck}^{DT} = \begin{bmatrix} -0.1783 & -0.0182 & 0.0167 & 1.6176e3 \end{bmatrix} \tag{5.24}$$

Then the check on the asymptotic stability is performed: considering C_q as the Cholesky factor of Q_o, the observability matrix `Mo = obsv(A,Cq)` is full rank. So the system is asymptotically stable.

5.4.2 Boost case

About the boost case, the design procedure is unchanged: considering the PWM relationships expressed in (5.14), the linearized matrices are computed as shown in (5.25). Because the system is controllable, the design proceeds exploiting again the additional integrative state q in order to ensure zero tracking error with a step reference.

$$\begin{aligned} A_\ell &= \begin{bmatrix} 1 - \frac{T_s}{R_{in}C_{in}} & 0 & -\frac{T_s}{C_{in}} \\ 0 & 1 - \frac{T_s}{R_{out}C_{out}} & -\frac{T_s(D_b-1)}{C_{out}} \\ \frac{T_s}{L} & \frac{T_s(D_b-1)}{L} & 1 \end{bmatrix} \quad B_\ell = \begin{bmatrix} 0 \\ -\frac{I_L K_3 T_s}{C_{out}} \\ \frac{V_{C_{out}} K_3 T_s}{L} \end{bmatrix} \\ C_\ell &= \begin{bmatrix} -\frac{1}{R_{in}} & 0 & 0 \end{bmatrix} \end{aligned} \tag{5.25}$$

The considered working point is $v_{in} = 8\,\mathrm{V}$, with maximum power transfer condition. The matrix Q and the scalar R are set and their values are summarized in table 5.5. The controller gain is computed using the `dlqr` command, obtaining as result (5.26).

Weight	q_1	q_2	q_3	q_i	R
Value	1e3	1	1	1e10	1e4

Table 5.5: Weights values for LQR control design, in DT boost case

$$K_{boost}^{DT} = \begin{bmatrix} -0.1708 & -0.0093 & 0.0328 & 756.7962 \end{bmatrix} \tag{5.26}$$

Finally, also the asymptotic stability is checked: considering C_q as the Cholesky factor of Q_o, the observability matrix is full rank and so the system is asymptotically stable.

5.4.3 Results

In order to measure the performances of the two controllers, four simulations are performed (shown in figures 5.8 - 5.11). All the results related to rise time, settling time and percentage overshoot are recollected in table 5.6. The tests for the points A and B are performed using the controller K_{boost}^{DT}, while the tests in points C and D are performed with the controller K_{buck}^{DT}. The working time chosen for the controller is $T_s = 1/f_s$, with $f_s = 75\,\text{kHz}$.

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	0.04	1.91	2.46
B	0.03	1.85	1.90
C	0.15	0.57	0.40
D	0.23	0.49	0.32

Table 5.6: Values of overshoot, rise time and settling time in each working point of interest for LQR controller in DT

Also for the DT case the performances are improved, resulting in a faster and more reactive system, with small overshoot in every working point. As a matter of facts, this analysis shows how a state control algorithm can be more effective than an output feedback: however, the LQR can be more difficult to be implemented in hardware. Indeed, the state can be not totally measured (as in this case) requiring the introduction of estimation algorithm that may slow down the time execution of the control law itself.

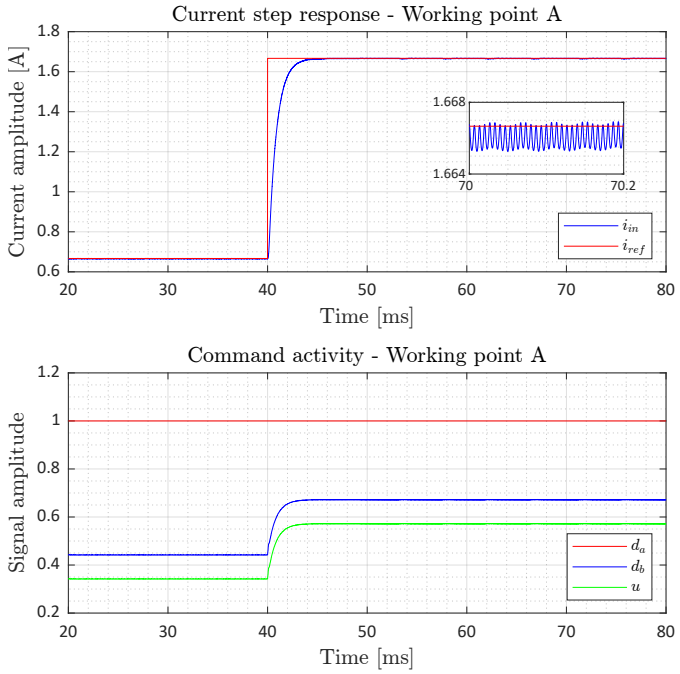


Figure 5.8: Simulation in working point A (boost) with LQR controller in DT

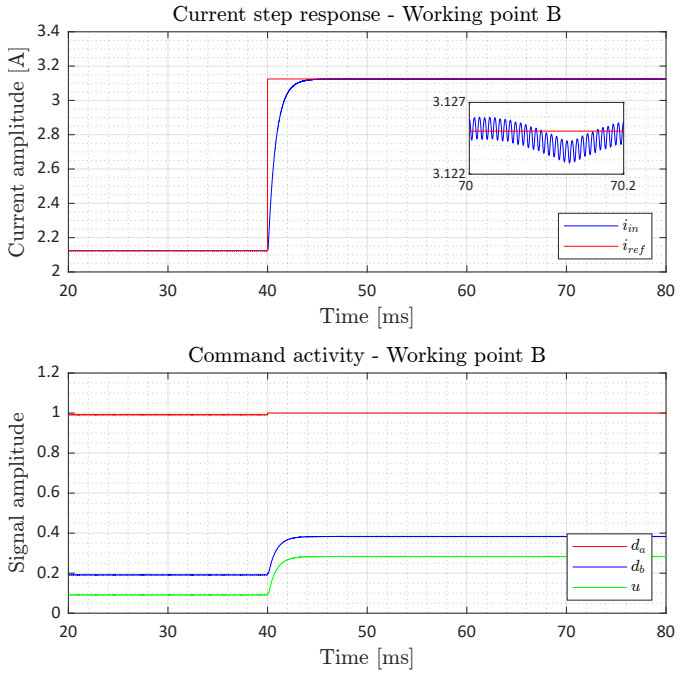


Figure 5.9: Simulation in working point B (boost) with LQR controller in DT

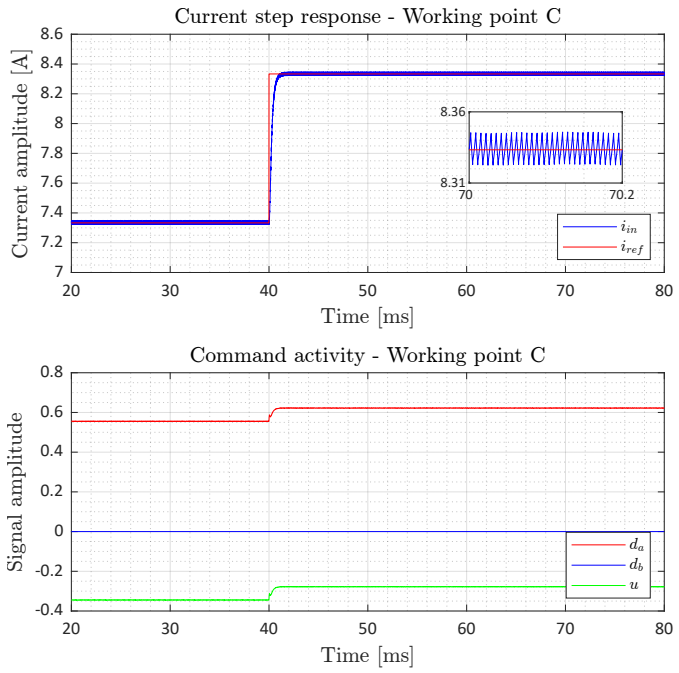


Figure 5.10: Simulation in working point C (buck) with LQR controller in DT

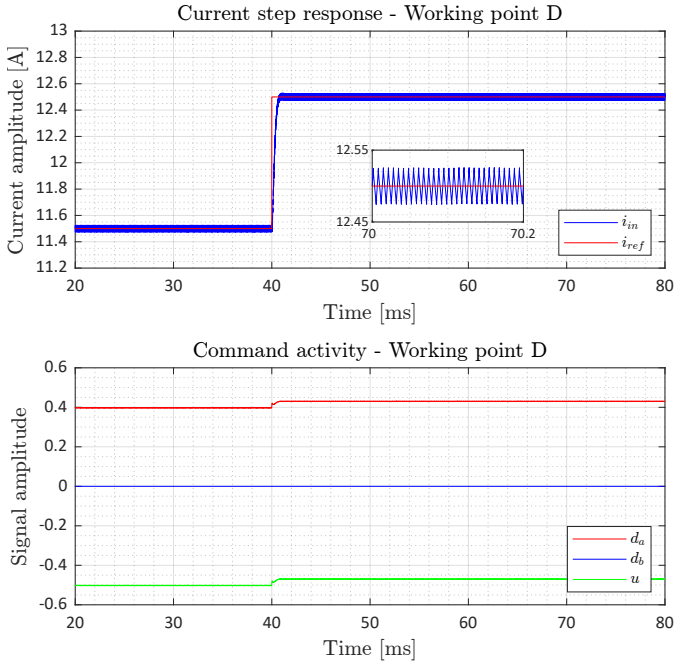


Figure 5.11: Simulation in working point D (buck) with LQR controller in DT

Chapter 6

Sliding mode control design

The sliding mode (**SM**) control is an advanced control technique that can be applied to different types of non-linear systems. It is based on the definition of a "sliding surface", that embodies the desired behavior of the system. The idea behind the design is to constraint the trajectory of the plant to follow the surface, ensuring robustness against imprecise knowledge of the plant and desired performance achievement. Like the LQR, it needs information from the state in order to work and, for some aspects, it is very similar to the feedback linearization technique.

In this chapter, after a brief introduction of the state of art related to this type of control law, the design process of the controller is described, paying attention to both CT and DT implementations. Finally, the step response of the system is studied for analysing the time performances of the controller.

6.1 State of art

The SM is a control technique that has gained popularity for the control of DC-DC converters: different studies have been carried out, explaining and outlining the features and the advantages of this type of control approach.

[33] develops a cascade controller with an inner current loop, exploiting a SM law, and an outer voltage loop regulated by a simple PI control. The NIBB topology used by the authors works only in the buck-boost area and so a single command is defined (i.e. both the switches have the same duty cycle). The sliding surface is defined in a very simple way, as the difference between the actual inductor current and the reference current provided by the external voltage loop: thus introducing a control law that does not need for tuning and defining a control system with a non-minimum phase structure. [34] implements the same control structure for the control of a NIBB converter used for the recharge of a Li-Ion battery of a laser guided vehicle. The SM control for the inner current loop is chosen for its robustness property and for the absence of gains to be tuned.

[35], instead, develops a modified SM control law, called pseudo sliding mode control, that operates at a fixed frequency. This type of technique is used in order to control a NIBB converter, exploiting a sliding surface as linear combination of the system state variables.

In [36] a more complex design is performed, where the SM technique is developed with two different methods, in order to estimate the uncertainties of the system. As first, the authors develop an adaptive SM control based on a state observer, then another type of SM controller is defined starting from a disturbance observer. Both the methods provide similar performances and good robustness against plant uncertainties.

[37] and [38] exploit the SM technique for controlling a buck converter in discrete time. [37] compares two types of DT implementations: the emulation design and the directly discrete design. For the first method, the controller is designed first in CT and then it is discretized, using a sample and hold circuit. The second method consists in developing the design directly in DT through a discrete approximation of the plant. Both the methods produce good results, even if the second one provides better performances.

[38], on the other hand, compares the emulation design with the DT design, carried using two different sliding surfaces: one with relative degree one and the other with relative degree two. The study outlines the improvements that can be obtained through the discrete design using a surface with relative degree equal to two.

6.2 Introduction

The design of a SM controller is completely based on the chosen sliding surface. The sliding surface is a state region where the system has a desired behavior: depending on how it is constructed, it can be exploited in order to obtain a zero tracking error in a finite time. The sliding surface is generally defined as:

$$S(t) = \{x \in \mathbb{R}^n : \sigma(x,t) = 0\}$$

where n is the order of the system and σ is the function that expresses the desired behavior. This function depends on the model used and on its relative degree, so it will change from boost to buck. About the control law, it is generally built in order to guarantee that the sliding surface is invariant and attractive:

- **INVARIANT**: if the trajectory of the system is on the surface, it has to remain on it. This property can be achieved imposing:

$$\dot{\sigma}(x,t) = 0 \tag{6.1}$$

- **ATTRACTIVE**: if the trajectory is not on the surface, it has to reach it. It can be ensured imposing:

$$\dot{\sigma}(x,t) \cdot \sigma(x,t) < 0 \tag{6.2}$$

In order to make S attractive and to satisfy the inequality of (6.2), a discontinuous term is introduced. Through the following equation:

$$\dot{\sigma}(x,t) = -s_1 \text{sign}(\sigma(x,t))$$

the inequality (6.2) is always satisfied and the attractiveness property is ensured. However, the discontinuous term may cause a phenomenon called chattering (i.e. the trajectory oscillates at high frequency around the sliding surface, causing an oscillation of the command input itself). In order to avoid this problem, a sigmoid function γ can be used instead of the sign function. A possible choice is the hyperbolic tangent, that ensures a smooth change:

$$\gamma(\eta\sigma) = \tanh(\eta\sigma)$$

The parameter η is a design parameter that can be tuned in order to reduce the chattering: the greater is η and the closer the sigmoid function is to the sign function. So it can be gradually reduced in order to reduce the chattering, if present.

6.3 Continuous time design

For the design of the controller in CT, it is important to pay attention to the definition of the sliding surface: indeed it depends on the mathematical model used for representing the system. As also described in chapter 5, the state space representation of the NIBB changes from boost to buck: in this case, the two representations are characterized by a different relative degree that makes not possible to use the same surface for both the working modes: this implies that two different controllers are needed.

6.3.1 Buck case

An affine representation of the system is exploited, in order to build the sliding surface:

$$\dot{x}(t) = f(x) + g(x)u$$
$$y = h(x)$$

The function f,g and h are smooth functions, derived considering the state space representation introduced in chapter 5 for the buck working mode (see (5.8) and (5.9)).

$$f(x) = \begin{bmatrix} \frac{v_{in}-v_{C_{in}}}{R_{in}C_{in}} - \frac{K_2 i_L}{C_{in}} \\ \frac{v_{out}-v_{C_{out}}}{R_{out}C_{out}} - \frac{i_L}{C_{out}} \\ \frac{K_2 v_{C_{in}}}{L} - \frac{v_{C_{out}}}{L} \end{bmatrix} \quad g(x) = \begin{bmatrix} -\frac{i_L}{C_{in}} \\ 0 \\ \frac{v_{C_{in}}}{L} \end{bmatrix} \tag{6.3}$$

$$h(x) = \frac{v_{in} - v_{C_{in}}}{R_{in}} \tag{6.4}$$

Observing how the output is defined ($y = i_{in}$), the relative degree of the system in buck case is one (deriving once the output, the command input u appears), with $a(x)$ and $b(x)$ defined as follows:

$$\begin{aligned} \dot{y}(t) &= a(x) + b(x)u(t) \\ a(x) &= \frac{v_{C_{in}} - v_{in} + i_L K_2 R_{in}}{C_{in} R_{in}^2} \\ b(x) &= \frac{i_L}{C_{in} R_{in}} \end{aligned} \tag{6.5}$$

These information are used for defining the function σ of the sliding surface. In order to guarantee a zero tracking error, σ is composed by two contributes: the tracking error and the integral of the tracking error itself.

$$\sigma_{buck}(t) = r(t) - y(t) + s_2 \int (r(t) - y(t))dt \tag{6.6}$$

In (6.6), r is the reference current i_r computed considering the maximum power transfer condition, y is the input current i_{in} and s_2 is the gain of the integral contribute. Applying the property of invariancy (6.1) and attractiveness (6.2), the command input u is computed:

$$\begin{aligned} u(t) &= \frac{1}{b(x)}(v(t) - a(x)) \\ v(t) &= \dot{r}(t) + s_2(r(t) - y(t)) + s_1 \tanh(\sigma_{buck}(t)\eta) \end{aligned} \tag{6.7}$$

In equation (6.7), $\dot{r}$ is assumed zero, because the reference changes slowly as a step and it can be approximated as a constant value. The tuning of the controller is then performed setting the values of s_1, s_2 and η, that are resumed in table 6.1. s_1 is related to the robustness of the plant: a very large value is needed also due to the model used, that does not take into account for the ESRs effect.

Parameter	s_1	s_2	η
Value	5e9	1	5e-4

Table 6.1: Parameters values for SM control design, in CT buck case

6.3.2 Boost case

For the boost working mode, the procedure is very similar, even if the functions f and g of the affine form are different. These functions are derived using the same equations (5.8) and (5.14).

$$f(x) = \begin{bmatrix} \frac{v_{in}-v_{C_{in}}}{R_{in}C_{in}} - \frac{i_L}{C_{in}} \\ \frac{v_{out}-v_{C_{out}}}{R_{out}C_{out}} - \frac{(1-K_4)i_L}{C_{out}} \\ \frac{v_{C_{in}}}{L} - \frac{(1-K_4)v_{C_{out}}}{L} \end{bmatrix} \quad g(x) = \begin{bmatrix} 0 \\ -\frac{i_L}{C_{out}} \\ \frac{v_{C_{out}}}{L} \end{bmatrix} \tag{6.8}$$

$h(x)$ is the same as (6.4). In this case, the relative degree of the system is two, meaning that the output has to be derived twice in order to make the command input u appear. $a(x)$ and $b(x)$ are defined as follows:

$$\ddot{y}(t) = a(x) + b(x)u(t)$$

$$\begin{aligned} a(x) &= \frac{v_{in} - v_{C_{in}}}{C_{in}^2 R_{in}^3} - \frac{i_L}{C_{in}^2 R_{in}^2} + \frac{v_{C_{in}} - v_{C_{out}}(1-K_4)}{C_{in}LR_{in}} \\ b(x) &= \frac{v_{C_{out}}}{C_{in}LR_{in}} \end{aligned} \tag{6.9}$$

Because the relative degree of the system is not one as in the buck case, the sliding surface definition changes:

$$\sigma_{boost}(t) = (\dot{r}(t) - \dot{y}(t)) + s_2(r(t) - y(t)) + s_3 \int (r(t) - y(t))dt \tag{6.10}$$

Now there are three contributes: the tracking error, its derivative and its integrative. Applying the property of invariancy (6.1) and attractiveness (6.2), the input command is computed as:

$$\begin{aligned} u(t) &= \frac{1}{b(x)}(v(t) - a(x)) \\ v(t) &= \ddot{r}(t) + s_3(r(t) - y(t)) + s_2(\dot{r}(t) - \dot{y}(t)) + s_1 \tanh(\sigma_{boost}(t)\eta) \\ \dot{y}(t) &= \frac{v_{C_{in}} - v_{in} + i_L R_{in}}{C_{in}R_{in}^2} \end{aligned} \tag{6.11}$$

In equation (6.11), $\dot{r}$ and $\ddot{r}$ are assumed zero, because the reference changes slowly as a step and it can be approximated as a constant value. The tuning of the controller is then performed setting the values of s_1, s_2, s_3 and η, that are resumed in table 6.2.

Parameter	s_1	s_2	s_3	η
Value	1e2	1e2	1e12	1e-6

Table 6.2: Parameters values for SM control design, in CT boost case

6.4 Results

The evaluation of the performances of the controller is carried out with four simulations, related to the same working points introduced in the previous chapters and shown in figures 6.1 - 6.4. The results obtained are collected in table 6.3. The tests are performed using the control law (6.7) for the buck working mode and the control law (6.11) for the boost working mode.

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	0.51	0.057	0.058
B	0.33	0.045	0.045
C	0.22	0.026	0.024
D	0.22	0.019	0.015

Table 6.3: Values of overshoot, rise time and settling time in each working point of interest for SM controller in CT

Observing the graphical and the numerical results, the SM controller provides a very fast response, characterized by rise times and settling times under 1 ms. However, the command activity presents very large oscillations, that could be an important issue in the hardware implementation, where very large changes in the command can lead to actuator issues.

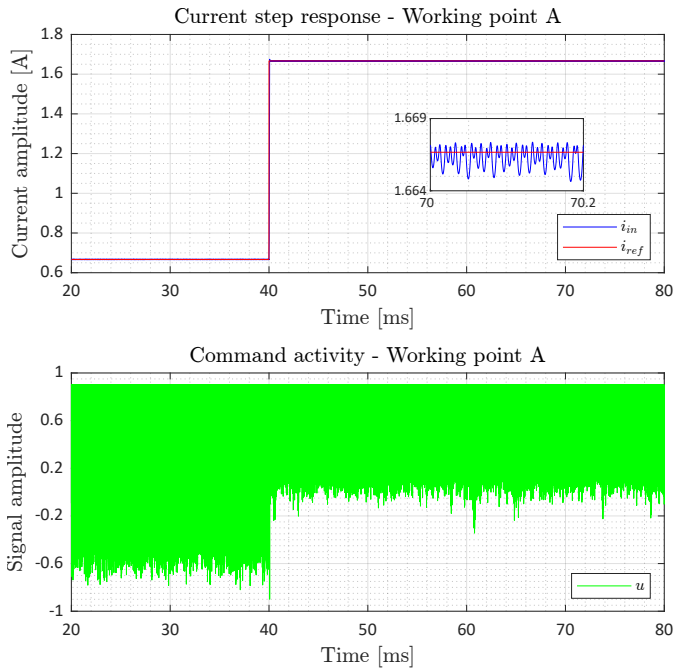


Figure 6.1: Simulation in working point A (boost) with SM controller in CT

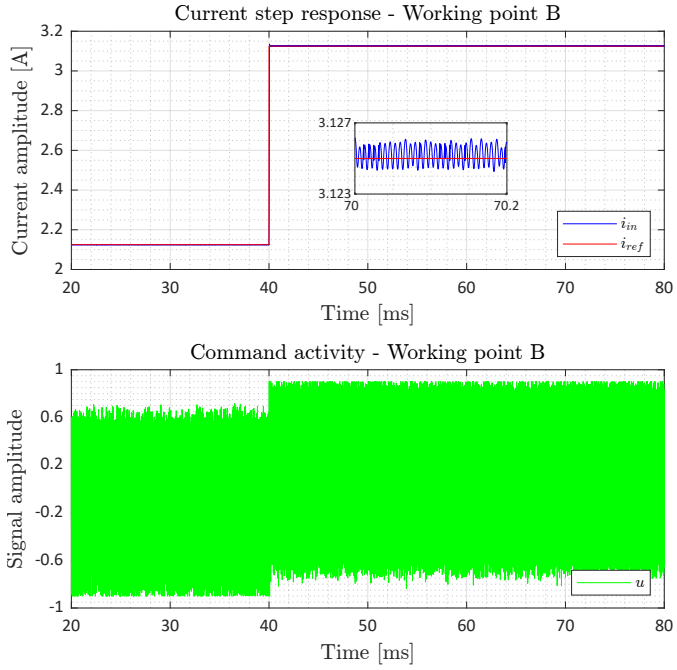


Figure 6.2: Simulation in working point B (boost) with SM controller in CT

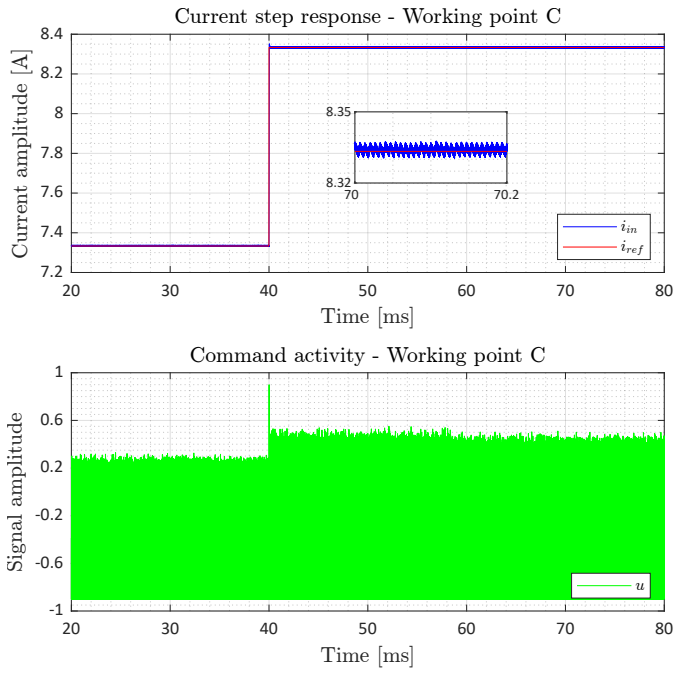


Figure 6.3: Simulation in working point C (buck) with SM controller in CT

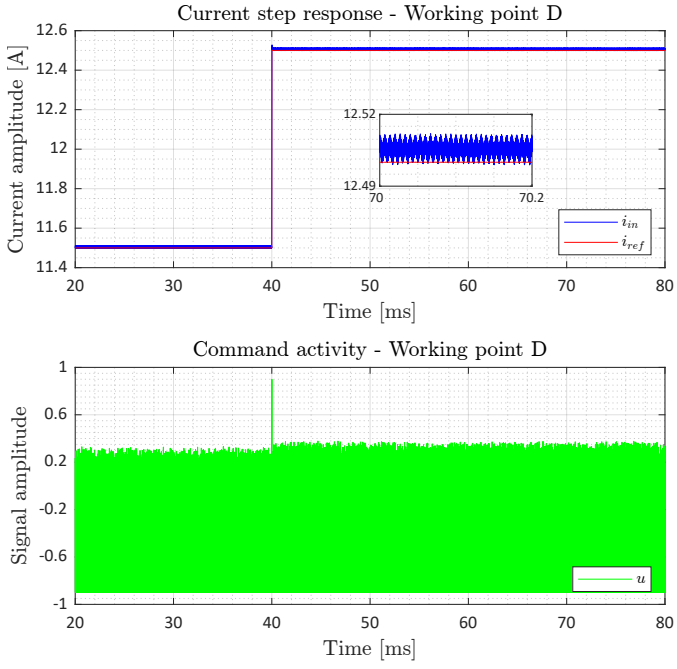


Figure 6.4: Simulation in working point D (buck) with SM controller in CT

6.5 Discrete time design

The DT implementation is provided through the emulation procedure. Differently from what described for the H_∞ control design, in this case the emulation consists in using directly the control law developed in CT, computing the command at each control working time T_s. The control time is set as $T_s = 1/f_s$, with $f_s = 30\,\text{kHz}$.

Therefore for the boost working area, the law described by (6.11) is considered, while for the buck working area, the system is controlled by means of (6.7). However, the parameters values are different and listed in table 6.4.

Parameter	s_1	s_2	s_3	η
Boost value	1e7	3e4	8e8	1e-5
Buck value	1e7	1e3	\	5e-5

Table 6.4: Parameters values for SM control design in DT

6.6 Results

Four simulations (shown in figures 6.5 - 6.8) are carried out, in order to observe the functioning of the controllers obtained with the emulation procedure. The results obtained are collected in table 6.3.

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	9.46	0.066	0.154
B	2.85	0.069	0.070
C	8.20	0.110	0.365
D	9.46	0.093	1.075

Table 6.5: Values of overshoot, rise time and settling time in each working point of interest for SM controller in DT

For the DT case, the rise time and the settling time remain close to the values computed for the CT case. However, the overshoot of the control variable is too large: the objective is to keep the overshoot close to zero, however, the simulations show overshoot values always greater than 2%. Moreover, the zero tracking error is not always achieved, in particular, if the simulation is carried in buck-boost region. As written in [38], this issue is related to the fact that the sliding surface is not reached and the state is inside a region of the sliding surface. This is a problem of robustness that, as also outlined in [39], affects the discrete implementation of the SM control (i.e. for the discrete time, the robustness of the system is not ensured).

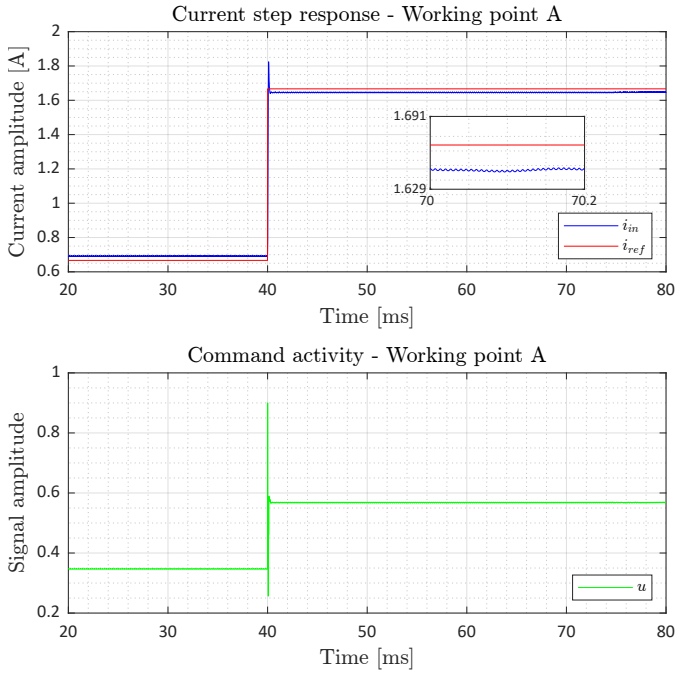


Figure 6.5: Simulation in working point A (boost) with SM controller in DT

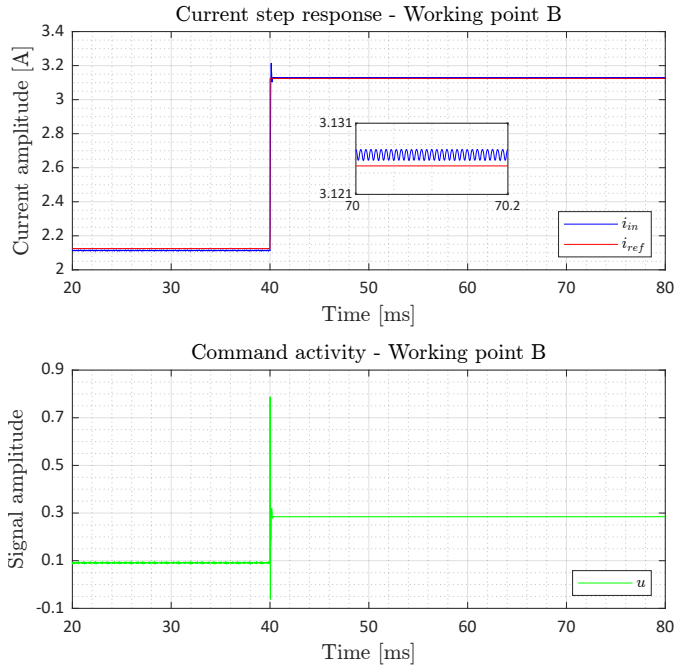


Figure 6.6: Simulation in working point B (boost) with SM controller in DT

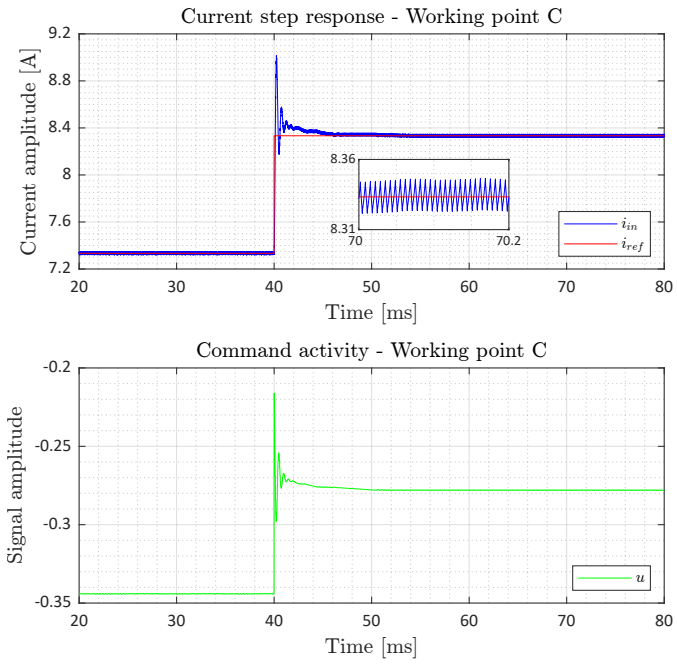


Figure 6.7: Simulation in working point C (buck) with SM controller in DT

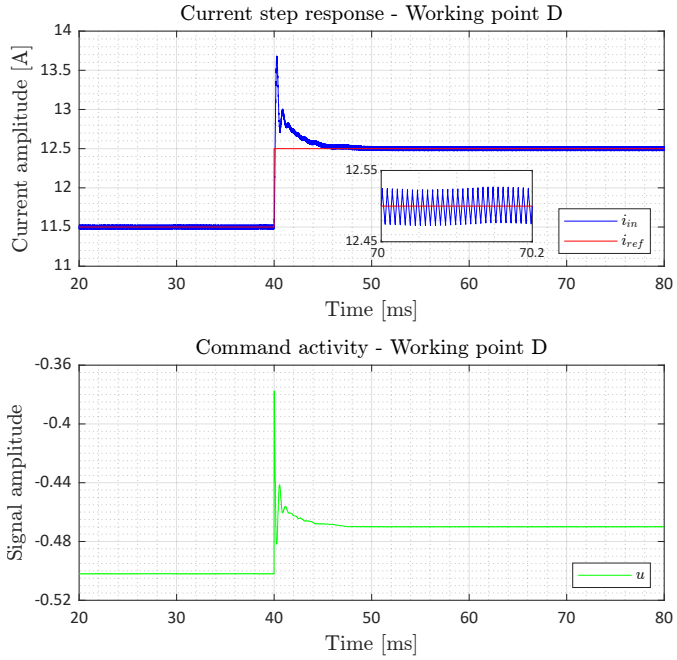


Figure 6.8: Simulation in working point D (buck) with SM controller in DT

Chapter 7

Fuzzy logic control design

All the control techniques introduced until now need a mathematical representation of the system in order to be designed. In chapter 2, two possible plant representations were introduced: a simple one, that is obtained by means of state space averaging, neglecting the effect of parasitic components, and a more complex one, in which also the ESRs' effects are taken into account. However, they remain approximations of the real behavior of the system. The state space averaging itself, that is used in order to derive both the models, is an approximation procedure, exploited for eliminating the time dependence of the system.

In this regard a Fuzzy Logic (**FL**) controller is able to overcome this issue: this type of controller is designed through the introduction of linguistic rules that mimic the human thinking. Because these rules depend only on the desired behavior of the system, no mathematical model is needed. This is a great advantage that makes this type of controller suitable for all the control problems, in which the plant model is too complex or not enough accurate.

In this chapter, after the state of art introduction, the FL theory is briefly outlined. Then the design procedure is provided, followed by the results analysis of the Simulink simulations.

7.1 State of art

Similarly to SM controllers, FL controllers have recently gained popularity in the field of DC-DC converters. Thanks to their flexibility and adaptability to every kind of complex system, they can be a good alternative to the classical PI controllers and to the more complex control techniques that relay on mathematical descriptions of the physical system.

[40] analyses the performances of an half-bridge LLC resonant converter when controlled with a traditional PID and with an FL controller. The FL controller is designed in order to take as inputs the tracking error of the output voltage, the

error difference and the error sum, resembling the functioning of a PID controller. The control action is defined trough simple linguistic rules related to the inputs previously introduced. The simulations carried out by the authors outline how the controller with FL improves the response of the system, that is faster than the response obtained applying the PID controller.

In [41] the author develops the design of a hybrid fuzzy/state-feedback control for a NIBB converter. The state-feedback aims to improve the stability and the time response of the system, while the fuzzy control structure is designed in order to ensure the tracking of the output voltage. The chosen input variables of the controller are the tracking error of the output voltage and the error difference. They are the inputs of two fuzzy controllers: one is dedicated to the computation of the normalized command variation while the other one regulates a gain updating factor that dynamically adjusts the output scale factor. So, the controller obtained is stable and robust, being able to self-tune.

[42] develops another interesting application of the FL for control of a boost converter. In this work, the authors combine the FL with the SM, in order to obtain a robust controller able to contain the chattering phenomenon, typical of the SM controllers. A PI control is designed as an outer voltage loop, that provides the reference for the inner current loop. The FL controller uses as linguistic variables the tracking error and the error difference, that are handled with a series of rules derived from the SM theory (the error is identified as the sliding surface of the system). The result is a controller robust against load and input variations, with good performances.

On the other hand, [43] develops a complex structure for the control of a NIBB converter, based on the Takagi-Sugeno fuzzy inference system. The control structure is composed by two PI controllers, implemented by means of FL and connected through a fuzzy switch, which selects the controller to be activated depending on the working mode. The controllers are modified in order to take into account also the oscillations related to the working point change. All the inference systems are based on rules derived from the application of a subtractive clustering algorithm.

Moreover, many studies about the combination of the FL with neural networks can be found in the literature: the main idea is to mix the decision making of the FL with the learning abilities of the neural networks, providing advance adaptive controllers. In [44] this concept is the basis for the design of an adaptive controller for a NIBB converter. The authors implement the procedure of the FL into the layers of a neural network, thus obtaining a controller that, after adequate training, is robust and able to provide good performances. The same design is employed for the control of a boost converter in [45]. Also [46] develops a similar idea, defining an FL controller in which both the rules and the membership functions are obtained through a parameter-learning algorithm. The algorithm is then implemented in an FPGA and exploited for controlling a buck-boost converter.

7.2 Introduction

Fuzzy controllers are a particular type of control algorithms based on the FL theory. The FL is a non-boolean logical system that tries to mimic the different shades of the natural language and the human-like thinking. For fuzzy controllers, the FL is the basic tool used for treating and evaluating linguistic variables through a linguistic control strategy. A detailed description of the FL can be found in [47] and [48]; here, instead, a brief introduction of the main concepts is given, for clarity sake.

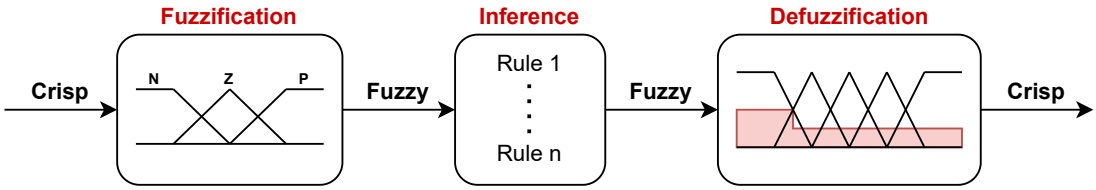


Figure 7.1: General description of the fuzzy process

As depicted in figure 7.1, the first step of a FL system is the fuzzification: this is a fundamental procedure, in order to apply the rules defined in the inference engine and so to exploit the linguistic control strategy. In the fuzzification, a variable that is initially crisp is translated in a fuzzy set, performing a scaling procedure. The scaling is necessary for increasing the sensibility around a set point and it is followed by a saturation on the universe of discourse. In this way, the scaled crisp variable is associated to a fuzzy set which is generally a singleton.

Then the fuzzy set obtained is elaborate by the inference system, exploiting a series of rules that are the base of the linguistic control strategy. The rules are characterized by an *if* part and a *then* part, that are related to a certain membership function obtained through the application of a compositional rule. The actual inputs are intersected with the membership function of the rule, obtaining the resulting fuzzy set. If more rules are fired, the result is obtained with the union of the resultant membership functions.

The fuzzy set obtained as result of the fuzzy inference engine has to be defuzzified and so translated into a suitable crisp variable, that is so the actual command. In this regard, the fuzzy set can be seen as the point of view of the human operator about the inputs of the system and, consequentially, the defuzzification is the execution of a suitable command based on the actual knowledge provided by the rules. The defuzzification procedure can be performed with many different methods: the most common is the center of gravity (**COG**), where the crisp variable is defined through the center of gravity of the resultant fuzzy set, obtaining a control action that is smooth and generally not extreme.

7.3 Discrete time design

The FL controller is implemented directly and only in discrete time. The whole design follows the procedure indicated in [49].

The first step into the design is the definition of the control variables. The most common solution is to choose the tracking error e, the variation of tracking error Δe and the variation of the input Δu. Instead of Δu, it is possible also to choose the command u, however the control of the command variation is simpler and it allows a finer control of the command itself.

$$r_n(k) = \frac{r(k)}{r(k)} = 1 \quad y_n(k) = \frac{y(k)}{r(k)} \tag{7.1}$$

The tracking error is computed modifying the reference and the output, as indicated in (7.1): this is necessary in order to simplify the tuning procedure of the controller, allowing to use only one controller for every working mode. The control variables' definitions are shown in (7.2), where $r = i_r$ is computed with maximum power transfer condition and $y = i_{in}$.

$$\begin{aligned} e(k) &= r_n(k) - y_n(k) \\ \Delta e(k) &= e(k) - e(k-1) \\ \Delta u(k) &= u(k) - u(k-1) \end{aligned} \tag{7.2}$$

This type of choice for the control variables define a structure that is commonly known as PD-Fuzzy controller, thus because the scaling factors introduced for e and Δe resemble the tuning parameters of a PD controller. The adopted structure is represented in figure 7.2.

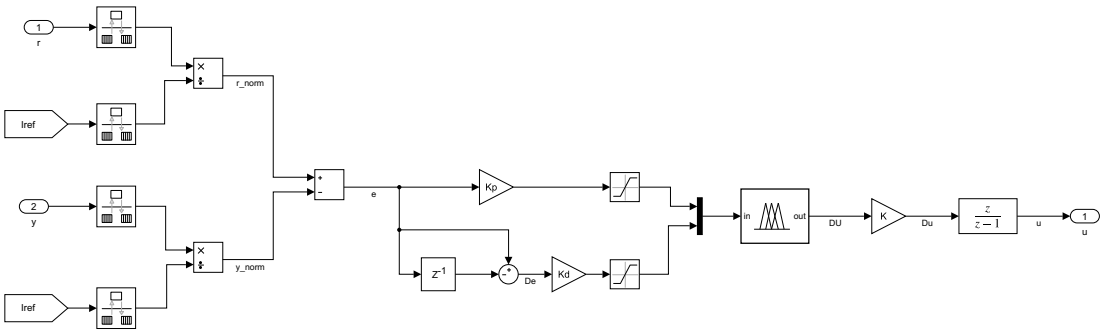


Figure 7.2: PD-Fuzzy controller structure

The scaling factors K, K_p and K_d are changed in order to reach a suitable sensibility that provides the desired performances. Also, the input variables are

followed by saturation blocks that avoid to have signals larger than the ranges defined for the corresponding universe of discourse in which the fuzzy sets are defined. The universes of discourse are set as follows:

$$U_e = [-1 \ 1]$$
$$U_{\Delta e} = [-1.5 \ 1.5]$$
$$U_{\Delta u} = [-0.1 \ 0.1]$$

Generally, thanks to the scaling factors, the universes of discourse can be set all equal: however, paying attention to the possible ranges of variation it is possible to obtain good performances with less tuning.

Then a set of seven terms [NB NM NS ZE PS PM PB] is defined in order to build the membership functions: each set is associated with a symmetric triangular-shape membership function. All the functions space from 0 to 1 in the universe of discourse previously introduced. Figures 7.3 - 7.5 show how the membership functions are built.

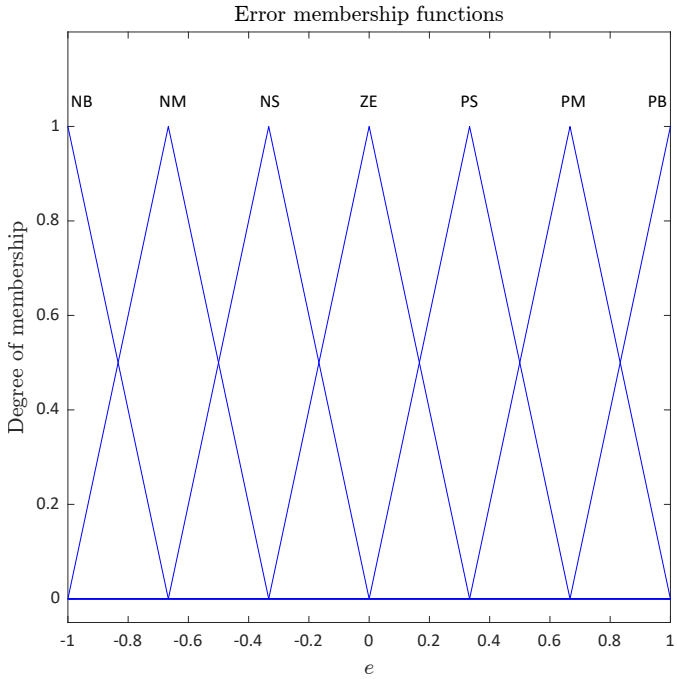


Figure 7.3: Membership functions related to the tracking error e

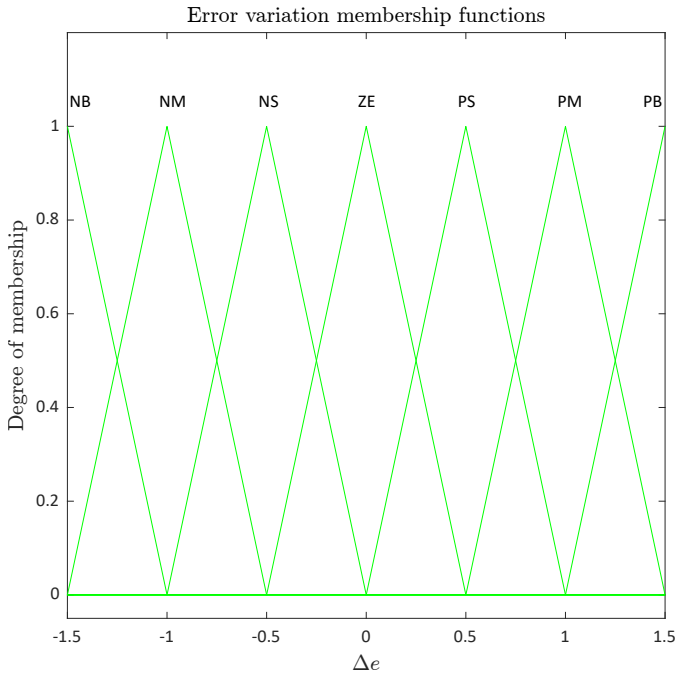


Figure 7.4: Membership functions related to the variation of tracking error Δe

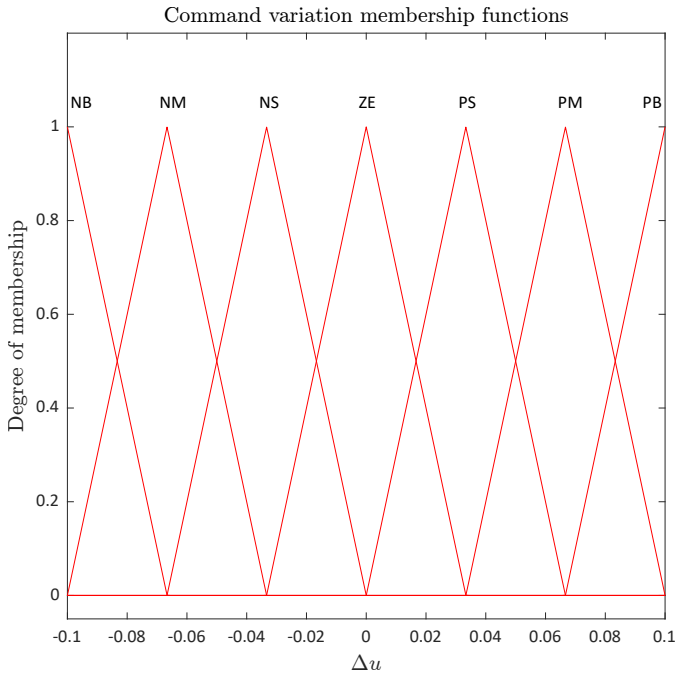


Figure 7.5: Membership functions related to the variation of command input Δu

Through the scaling factors and the saturation blocks, the signals E and ΔE are computed and given as inputs of the fuzzy inference engine. The chosen engine is a Mamdani type, with *min* as intersection operator and *max* as union operator. The inference process uses a rule base of 49 rules, in order to correctly implement the linguistic control strategy. Each rule is derived through the expert knowledge, observing the behavior of a general second order step response. The rule base is resumed in table 7.1, that is also an exemplification of the control strategy.

Δe\e	**NB**	**NM**	**NS**	**ZE**	**PS**	**PM**	**PB**
PB	ZE	PS	PS	PM	PM	PB	PB
PM	NS	ZE	PS	PM	PM	PB	PB
PS	NM	NS	ZE	PS	PS	PB	PB
ZE	NB	NM	NS	ZE	PS	PM	PB
NS	NB	NB	NS	NS	ZE	PS	PM
NM	NB	NB	NM	NS	NS	ZE	PS
NB	NB	NB	NM	NM	NS	NS	ZE

Table 7.1: Rule base of the fuzzy inference system

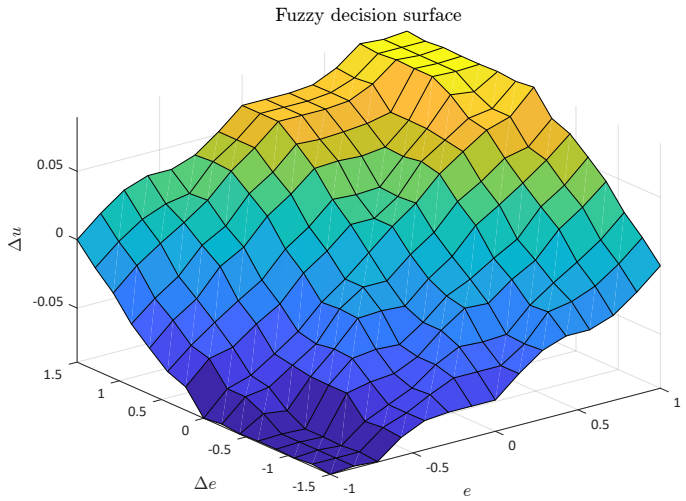


Figure 7.6: Decision surface of the fuzzy inference engine

The defuzzification is provided with the COG algorithm and the result is the command variation ΔU, that is then scaled by the factor K. Finally the command input is provided:

$$\Delta u(k) = u(k) - u(k-1) = u(k) - u(k)z^{-1}$$

$$u(k) = \frac{z}{z-1}\Delta u(k)$$

The values for the scaling factors are resumed in table 7.2.

Factor	K_p	K_d	K
Value	5	100	0.1

Table 7.2: Values of the scaling factors used in the PD-Fuzzy controller

7.4 Results

The simulations in points A,B,C and D are provided in order to observe the capacities of the FL controller: they are presented in figures 7.7 - 7.10, whit all the meaningful quantities collected in table 7.3. The working frequency chosen for the controller is $f_s = 30\,\text{kHz}$.

W.P.	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
A	2.86	1.21	2.54
B	2.27	1.26	2.92
C	0.60	1.75	1.23
D	0.42	1.27	0.72

Table 7.3: Values of overshoot, rise time and settling time in each working point of interest for Fuzzy controller in DT

The results obtained are interesting: the controller shows better performances than the ones related to the linear controllers previously introduced. The command signal has no significant oscillations and, considering that the velocity of the system is close to the velocity obtained with the SM, it appears as the best choice for controlling the NIBB converter. Even if the overshoot is slightly greater than 2% in boost working mode, the rise time and the settling time are always small than 3 ms, showing a very fast step response. Moreover, differently from the SM control, the tracking is always ensured, even in buck-boost mode. Furthermore, another greater advantage is the fact that a mathematical model is not necessary, making the design procedure simple and fast.

The main drawback of this controller is the absence of a theoretical foundation, that cannot ensure the robustness of the system. As a matter of facts, the robustness can be checked only by means of experience, because no theoretical result can be provided.

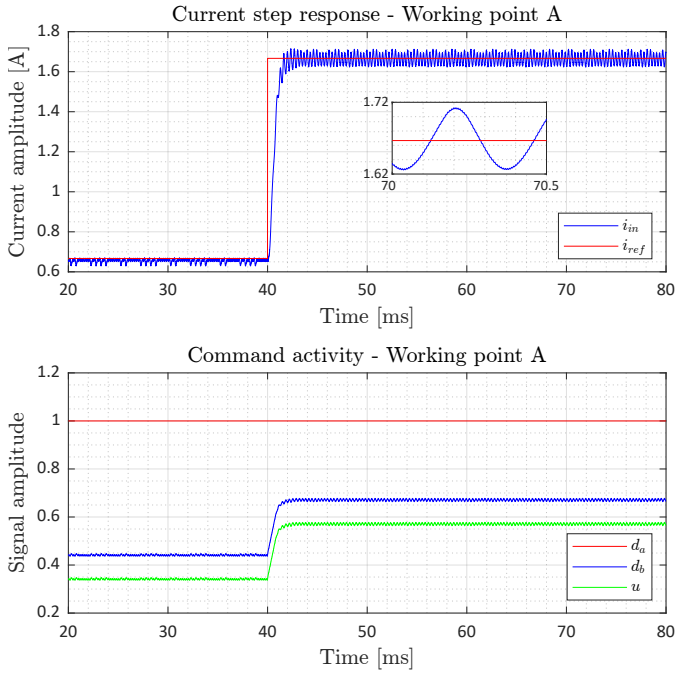


Figure 7.7: Simulation in working point A (boost) with FL controller in DT

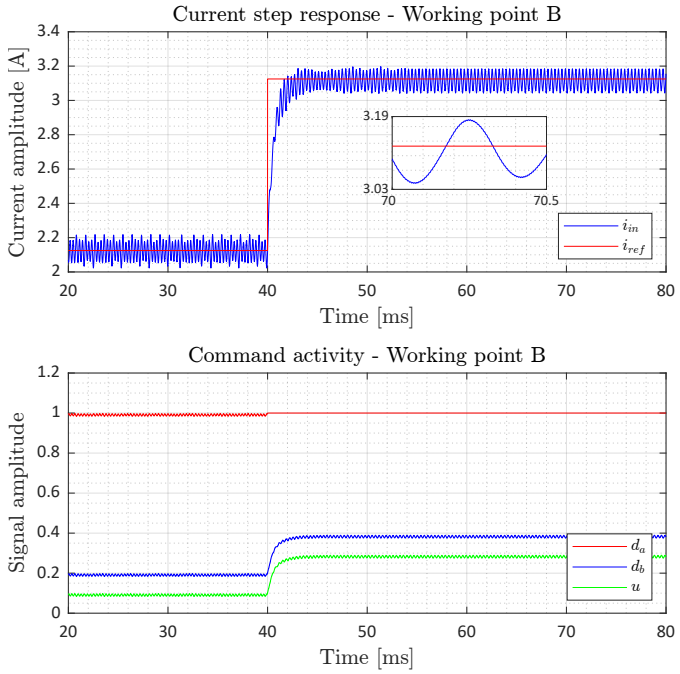


Figure 7.8: Simulation in working point B (boost) with FL controller in DT

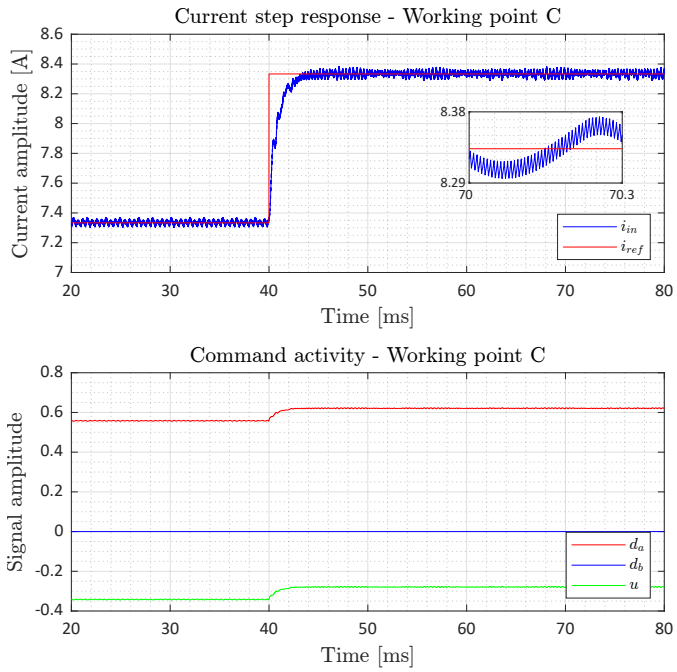


Figure 7.9: Simulation in working point C (buck) with FL controller in DT

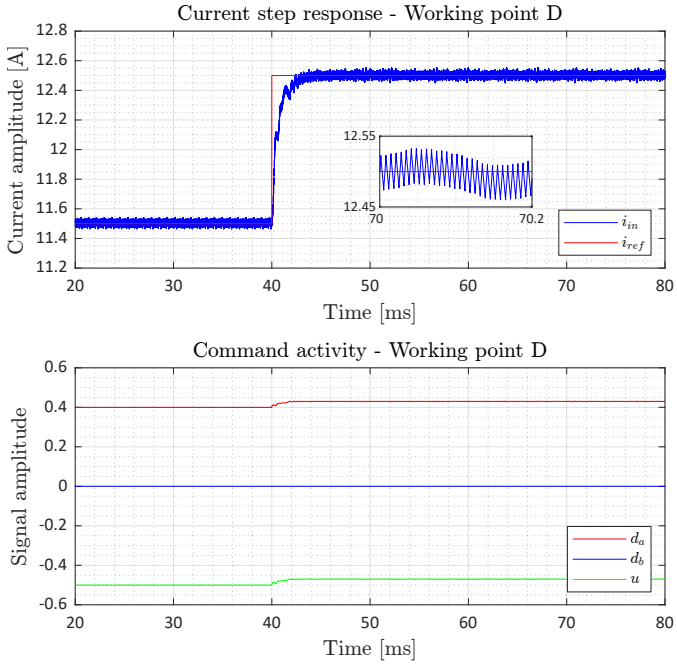


Figure 7.10: Simulation in working point D (buck) with FL controller in DT

Chapter 8

Filtering and state estimation

In the previous chapters, different control techniques have been described in order to correctly drive the switches of the DC-DC NIBB convert. Depending on the control algorithm developed, the control variable used may be the input current alone (output feedback) or the total state of the system (state feedback). However, in many practical cases, to use the state of the system might be difficult: for a DC-DC converter, for example, the measure of the inductor current might be critical, due to the price of the sensors used, that are in general very expensive, and to the accuracy of the measure itself, that is usually affected by noise. Also, there might be some problems with the input current measurement itself, that may be affected by important noises which can degrade the performances of the controller.

For these reasons, an Extended Kalman Filter (**EKF**) is employed. The EKF allows not only to estimate the inductor current, that is fundamental for the state feedback controllers, but also to provide an important attenuation of the noises related to the available measures.

This chapter provides a complete description for the EKF implementation. After introducing the state of art related to the application of the EKF in the DC-DC converters field, a classical EKF is implemented. Then, an innovative structure with two EKF blocks in cascade is proposed and compared with the classical EKF implementation.

8.1 State of art

The main applications of the EKF in the DC-DC converters field, described in the literature, focus mainly on simple types of converters, such as buck and boost converters.

For example, [50] and [51] apply an EKF to a simple buck converter, modeled

with state space averaging. [51] proposes an hybrid EKF that can operate both in CCM and DCM, being able to estimate not only the state, but also the load variations and the operation mode. This study shows interesting results, demonstrating the accuracy reached with the estimation and outlining the improvements that an EKF implementation can lead in terms of control. [50] implements an EKF algorithm for improving the performances of a buck converter and for estimating the inductor current used for the implementation of a simple predictive control.

[52] and [53], instead, develop an improved EKF for a DC-DC boost converter. After applying the state space averaging in order to obtain a suitable model for the prediction, they design a load variation effect elimination module. In this way the estimate is not corrupted by any possible change of the load of the converter. In both the works the objective is to measure the inductor current without using any expensive sensor and to improve the measures provided by the other physical sensors.

8.2 Single Kalman filter

In order to build an EKF, a discrete time representation of the system is needed and it can be derived exploiting the Forward Euler approximation as done in chapter 5. In this particular case, the vector state is composed not only by the inductor current and the capacitors voltages but also by the input/output voltages and the input/output resistors. As a matter of facts, in the real implementation of the circuit these values are not completely known. The input and the output voltages are not measured, as well as the input and the output resistors.

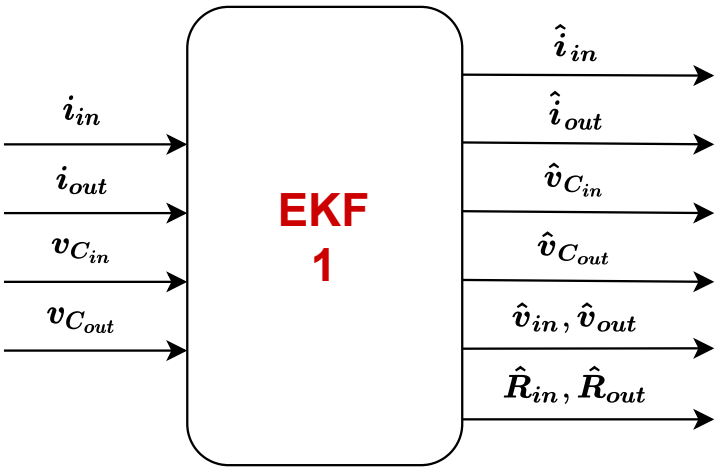


Figure 8.1: Single EKF general scheme

So, in order to have a correct estimation, these quantities are introduced as states of the system itself and estimated:

$$x(k) = \begin{bmatrix} v_{C_{in}}(k) & v_{C_{out}}(k) & i_L(k) & v_{in}(k) & v_{out}(k) & R_{in}(k) & R_{out}(k) \end{bmatrix}^T$$

Assuming that the values of v_{in}, v_{out}, R_{in} and R_{out} change very slowly in time, the function f is defined as $f(x(k), u(k)) = x(k+1)$, where each entry of the expression is expressed as follows:

$$v_{C_{in}}(k+1) = v_{C_{in}}(k)\left(1 - \frac{T_s}{R_{in}(k)C_{in}}\right) - \frac{d_a(k)i_L(k)T_s}{C_{in}} + \frac{T_s v_{in}(k)}{R_{in}(k)C_{in}}$$

$$v_{C_{out}}(k+1) = v_{C_{out}}(k)\left(1 - \frac{T_s}{R_{out}(k)C_{out}}\right) - \frac{(1-d_b(k))i_L(k)T_s}{C_{out}} + \frac{T_s v_{out}(k)}{R_{out}(k)C_{out}}$$

$$i_L(k+1) = \frac{d_a(t)v_{C_{in}}(k)T_s}{L} - \frac{(1-d_b(t))v_{C_{out}}(k)T_s}{L} + i_L(k)$$

$$v_{in}(k+1) = v_{in}(k)$$

$$v_{out}(k+1) = v_{out}(k)$$

$$R_{in}(k+1) = R_{in}(k)$$

$$R_{out}(k+1) = R_{out}(k)$$

The function h expresses known quantities that are measured. In this application, the available measured quantities are the voltages on the capacitors and the input and output currents. h is so defined as $h(x(k)) = z(k)$, where i_{in} and i_{out} are expressed as functions of the state itself. Thus allowing to have enough information about the unmeasured quantities that the Kalman filter has to estimate.

$$z(k) = \begin{bmatrix} i_{in}(k) & i_{out}(k) & v_{C_{in}}(k) & v_{C_{out}}(k) \end{bmatrix}^T$$

$$i_{in}(k) = \frac{v_{in}(k) - v_{C_{in}}(k)}{R_{in}(k)} \qquad i_{out}(k) = \frac{v_{C_{out}}(k) - v_{out}(k)}{R_{out}(k)}$$

After defining the functions f and h, the matrices $\hat{A}$ and $\hat{C}$ are obtained through the linearization procedure. Because the matrices are too big for being represented, only the non-null entries are shown.

Equation (8.1) shows the non-null entries of $\hat{A}$, that is then evaluated in $\hat{x}(k|k)$.

$$\begin{aligned}
&\hat{A}(1,1) = -\frac{T_s - C_{in}R_{in}(k)}{C_{in}R_{in}(k)} && \hat{A}(1,3) = -\frac{d_a(k)}{C_{in}} \\
&\hat{A}(1,4) = \frac{T_s}{C_{in}R_{in}(k)} && \hat{A}(1,6) = \frac{T_s(v_{C_{in}}(k) - v_{in}(k))}{C_{in}R_{in}(k)^2} \\
&\hat{A}(2,2) = 1 - \frac{T_s}{C_{out}R_{out}(k)} && \hat{A}(2,3) = -\frac{T_s(d_b(k) - 1)}{C_{out}} \\
&\hat{A}(2,5) = \frac{T_s}{C_{out}R_{out}(k)} && \hat{A}(2,7) = \frac{T_s(v_{C_{out}}(k) - v_{out}(k))}{C_{out}R_{out}(k)^2} \\
&\hat{A}(3,1) = \frac{T_s d_a(k)}{L} && \hat{A}(3,2) = \frac{T_s(d_b(k) - 1)}{L} \\
&\hat{A}(3,3) = 1 && \hat{A}(4,4) = 1 \\
&\hat{A}(5,5) = 1 && \hat{A}(6,6) = 1 \\
&\hat{A}(7,7) = 1
\end{aligned} \tag{8.1}$$

As well, equation (8.2) shows the non-null entries of $\hat{C}$, that is then evaluated in $\hat{x}(k|k-1)$.

$$\begin{aligned}
&\hat{C}(1,1) = -\frac{1}{R_{in}(k)} && \hat{C}(1,4) = \frac{1}{R_{in}(k)} \\
&\hat{C}(1,6) = \frac{v_{C_{in}}(k) - v_{in}(k)}{R_{in}(k)^2} && \hat{C}(2,2) = \frac{1}{R_{out}(k)} \\
&\hat{C}(2,5) = -\frac{1}{R_{out}(k)} && \hat{C}(2,7) = \frac{v_{C_{out}}(k) - v_{out}(k)}{R_{out}(k)^2} \\
&\hat{C}(3,1) = 1 && \hat{C}(4,2) = 1
\end{aligned} \tag{8.2}$$

With the linearization matrices, the algorithm can be completely implemented. The last needed operations are the initialization and the tuning of the variance matrices. First of all, a first prediction $\hat{x}(1|0)$ has to be set: this prediction can be seen as an estimate of the first state of the system and can be defined in a reasonable way from what is known of the system. For example, the first state of v_{out} and $v_{C_{out}}$ can be set equal to 12 V, that is the nominal voltage of the battery to which the NIBB converter is connected. Because no "a priori" information is given about v_{in} and $v_{C_{in}}$, they are set to 0. While, about R_{in} and R_{out}, it is possible to use the related data-sheet values. Then, the variance matrix P_1 related to $\hat{x}(1|0)$ has

to be chosen: generally it is set as an identity matrix (in this case I_{7x7}), however, for this application, the last three entries are set to 0.

$$\hat{x}(1|0) = \begin{bmatrix} 0 & 12 & 0 & 0 & 12 & 2.4 & 22.5e-3 \end{bmatrix}^T \qquad P_1 = \text{diag}(1,1,1,1,0,0,0)$$

Finally, the tuning of the EKF is provided through the matrices V_1 and V_2. These matrices can be seen as the degree of confidence related to the model and to the measurements respectively. Generally V_2 can be set as the variance matrix of the noise that affects the measures (if it can be extracted) while V_1 is tuned by trial and error, observing the results of the simulations and the RMSE values. For simplicity, V_1 and V_2 are defined as diagonal matrices.

$$V_1 = \text{diag}(1,0.1,500,0,0,0,0) \qquad V_2 = \text{diag}(0.1,0.1,0.1,0.1)$$

8.2.1 Results

In order to tune and test properly the EKF, a simulation is provided. The simulations are performed in the working points A,B,C and D, adding to the measures a random Gaussian noise, with 0.03 variance and 0 mean. The system is controlled with a PI controller, as designed in chapter 3 for DT, thus in order to impose a realistic command input. The results are collected in table 8.1, with figures 8.2 and 8.3 representing the states of the system obtained with the simulations in point B and C. Moreover, in order to reduce the effect of the transient phase, the first 20 ms (corresponding to the first 600 signal samples) are neglected for the computation of the RMSE. The sampling frequency chosen is $f_s = 30kHz$.

$var \backslash W.P.$	A	B	C	D
$v_{C_{in}}$	0.0814	0.0806	0.0801	0.0802
$v_{C_{out}}$	0.0040	0.0040	0.0040	0.0040
i_L	1.0374	0.6009	0.6605	1.0543
i_{in}	0.0339	0.0337	0.0345	0.0360
i_{out}	0.1765	0.1766	0.1767	0.1767

Table 8.1: Values of RMSE of the single EKF for four different working points, neglecting the first 600 samples

As it is possible to observe, the single EKF works well, being able to reduce the effects of the noise and to estimate the states with enough accuracy. However, the estimate of the inductor current is characterized by a small bias that might be a problem in the implementation of a state-feedback controller.

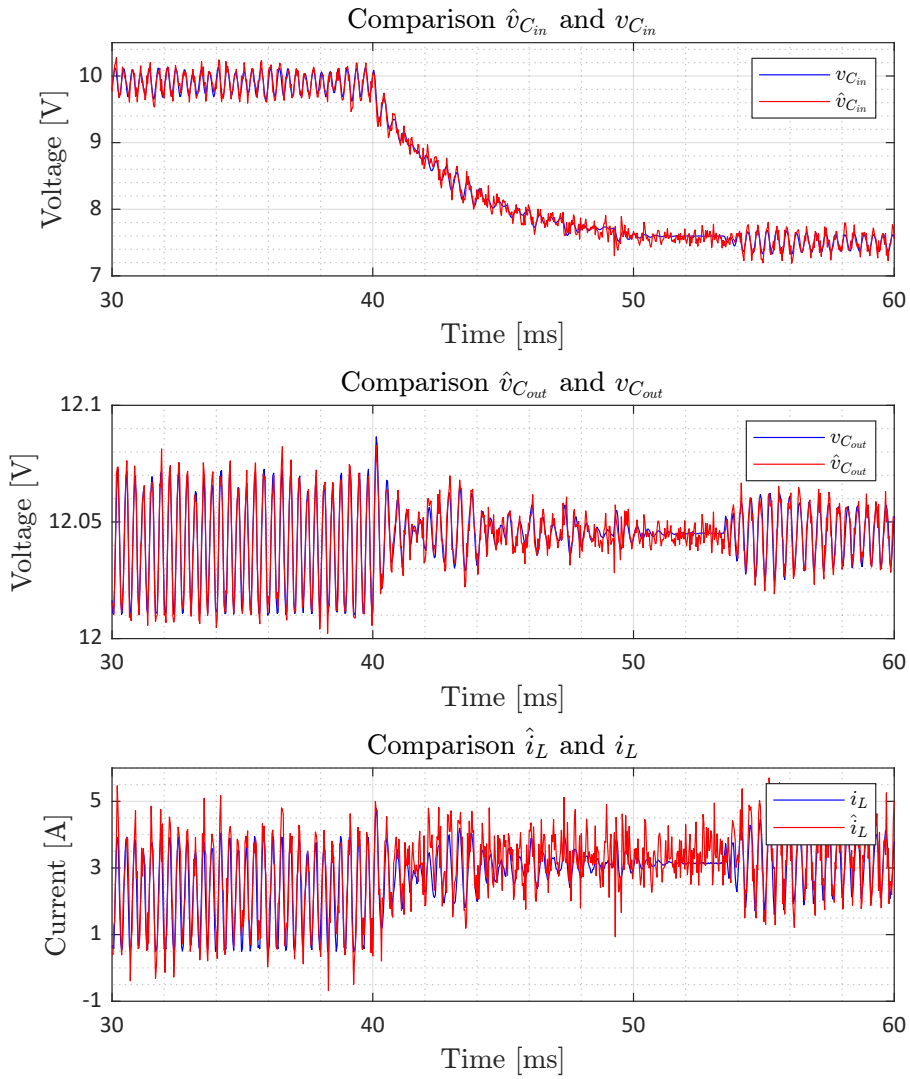


Figure 8.2: Simulation in working point B (boost) with single EKF

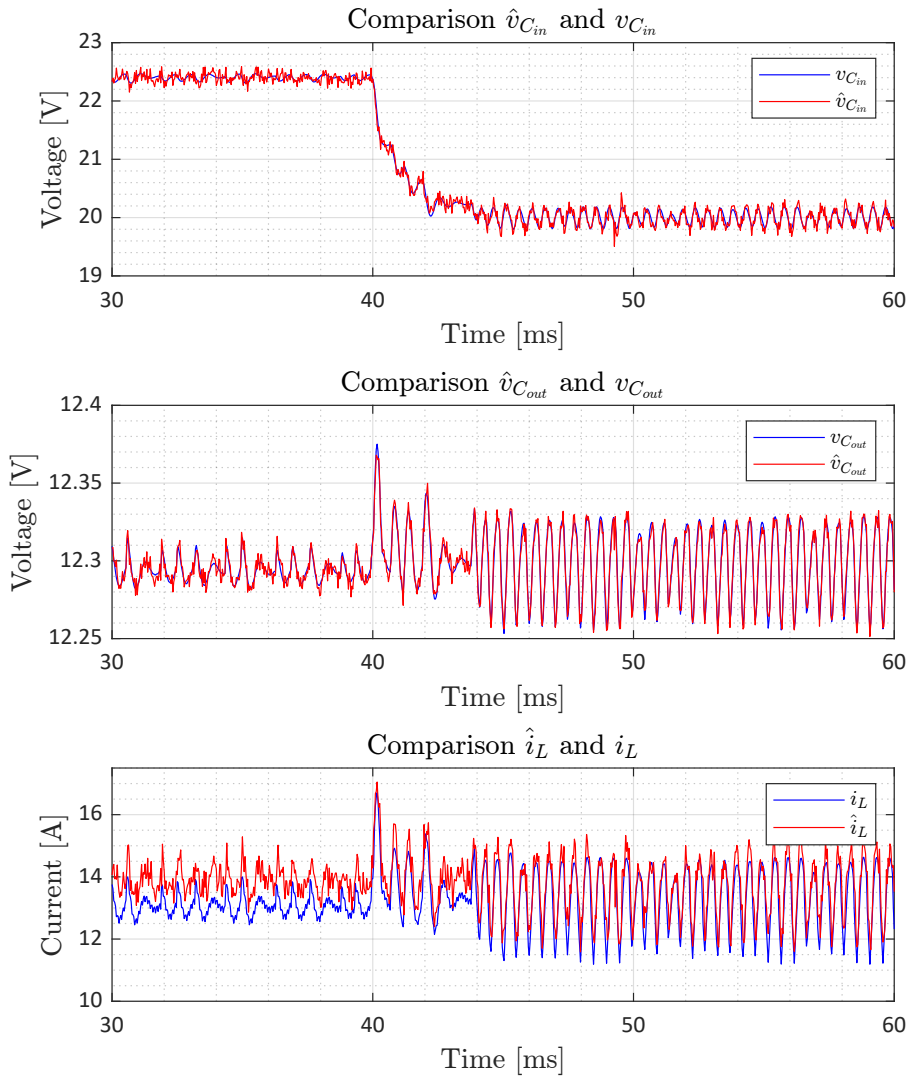


Figure 8.3: Simulation in working point C (buck) with single EKF

8.3 Cascade Kalman filter

The cascade filter is a particular application of the Kalman filter, where two EKFs with different equations are implemented.

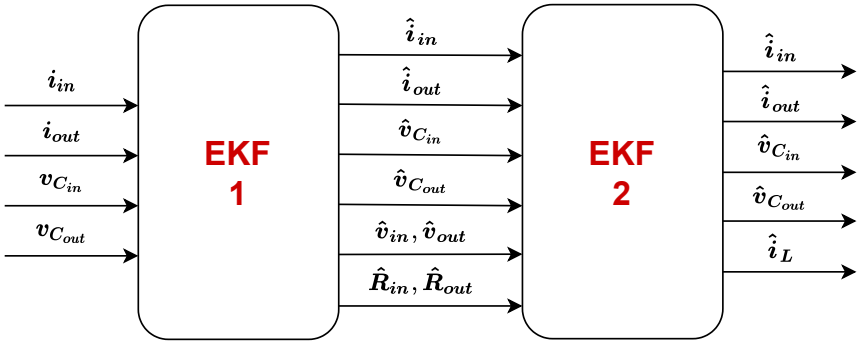


Figure 8.4: Cascade EKF general scheme

The first EKF is exactly equal to the filter built in the previous section: its task is to give an initial estimation of the state and to clean up the measures from the added noise. The second filter, instead, refines the estimates with the implementation of another model. In particular, this structure allows to reduce the bias of the inductor current, that may be an important problem for the state-feedback control.

The definition of the second EKF is similar to the first one: the equations of the state do not change, however v_{in}, v_{out}, R_{in} and R_{out} are no more considered as states. As a matter of facts, after the first estimation from the first Kalman filter, they are considered as known for the second filter. The main change for the second EKF is related to the measures expression. The measured quantities are always the same, however, the expressions of i_{in} and i_{out} are different: instead of using the Kirchhoff law, the relationships of the big signal analysis are implemented.

$$i_{in}(k) \approx d_a(k) i_L(k)$$
$$i_{out}(k) \approx (1 - d_b(k)) i_L(k)$$

This change leads to a variation of the matrix $\hat{C}$. For clarity sake, both matrices $\hat{A}$ and $\hat{C}$ related to the second EKF are represented in equations (8.3) and (8.4).

In this case, with three states, matrix $\hat{A}$ is a 3x3 matrix:

$$\begin{aligned}
&\hat{A}(1,1) = -\frac{T_s - C_{in}R_{in}(k)}{C_{in}R_{in}(k)} \quad \hat{A}(1,3) = -\frac{d_a(k)}{C_{in}} \\
&\hat{A}(2,2) = 1 - \frac{T_s}{C_{out}R_{out}(k)} \quad \hat{A}(2,3) = -\frac{T_s(d_b(k)-1)}{C_{out}} \\
&\hat{A}(3,1) = \frac{T_s d_a(k)}{L} \quad \hat{A}(3,2) = \frac{T_s(d_b(k)-1)}{L} \\
&\hat{A}(3,3) = 1
\end{aligned} \tag{8.3}$$

while matrix $\hat{C}$ is a 4x3 matrix, because all the previous measures are considered.

$$\begin{aligned}
&\hat{C}(1,3) = d_a(k) \quad \hat{C}(2,3) = 1 - d_b(k) \\
&\hat{C}(3,1) = 1 \quad \hat{C}(4,2) = 1
\end{aligned} \tag{8.4}$$

The tuning and the initialization are provided exactly in the same way previously described for the first EKF:

- **Initialization**:

$$\hat{x}(1|0) = \begin{bmatrix} 0 & 12 & 0 \end{bmatrix}^T \qquad P_1 = \text{diag}(0.1,1,10)$$

- **Tuning**:

$$V_1 = \text{diag}(1,12,1) \qquad V_2 = \text{diag}(0.1,0.1,0.1,0,1)$$

8.3.1 Results

The cascade EKF provides results that are close to the ones obtained with the single EKF. However, the bias of the inductor current is significantly reduced. The filter is tested with the same previous simulations and all the RMSE values are resumed in table 8.2 (neglecting the first 600 samples). Also, figures 8.5 and 8.6 show the simulations in point B and C ,that can be compared with the previous ones, in figures 8.2 and 8.3. As a matter of facts, these tests proof that combining different information is possible to reach better results in the estimation of the inductor current; however it has to be taken into account the complexity level of this filter, that, indeed, takes a larger computational time.

$var \backslash W.P.$	A	B	C	D
$v_{C_{in}}$	0.0750	0.0747	0.0748	0.0751
$v_{C_{out}}$	0.0041	0.0041	0.0045	0.0050
i_L	0.4042	0.6008	0.3016	0.3704
i_{in}	0.0313	0.0311	0.0319	0.0332
i_{out}	0.1802	0.1805	0.1198	0.2226

Table 8.2: Values of RMSE of the cascaded EKF for four different working points, neglecting the first 600 samples

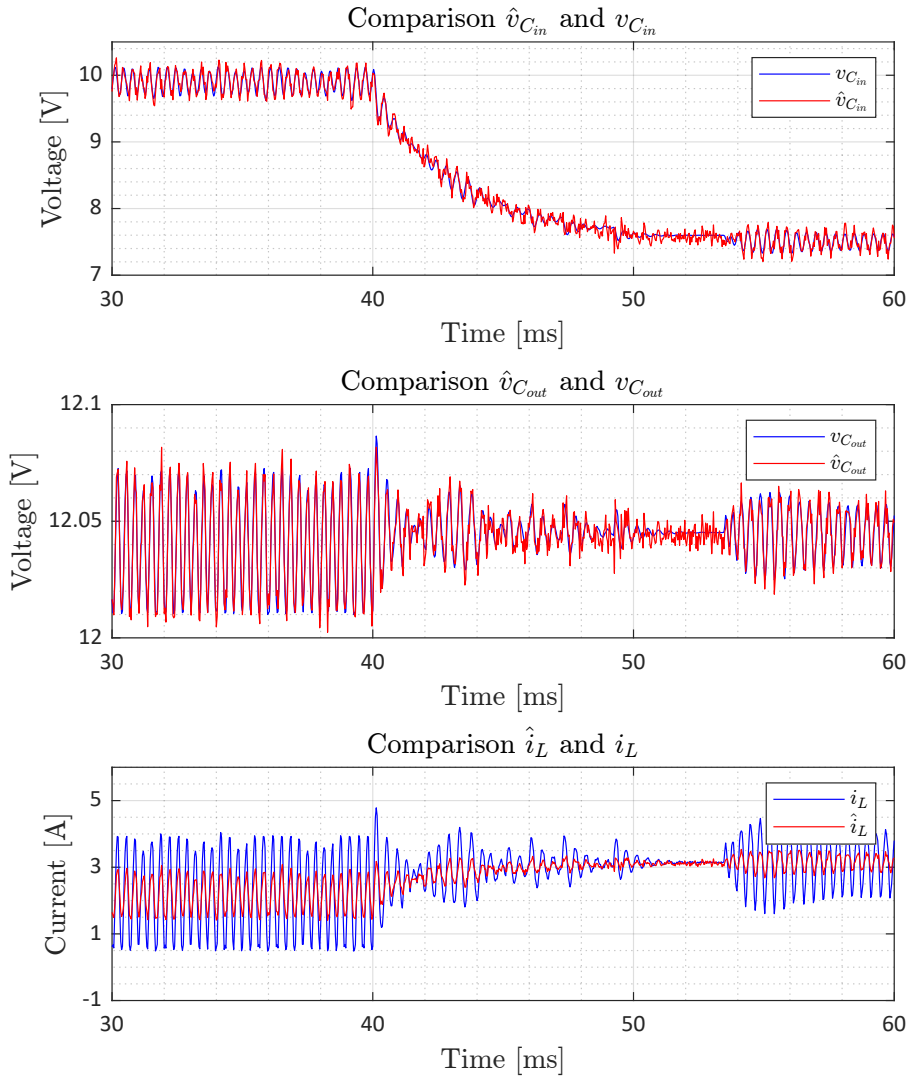


Figure 8.5: Simulation in working point B (boost) with cascade EKF

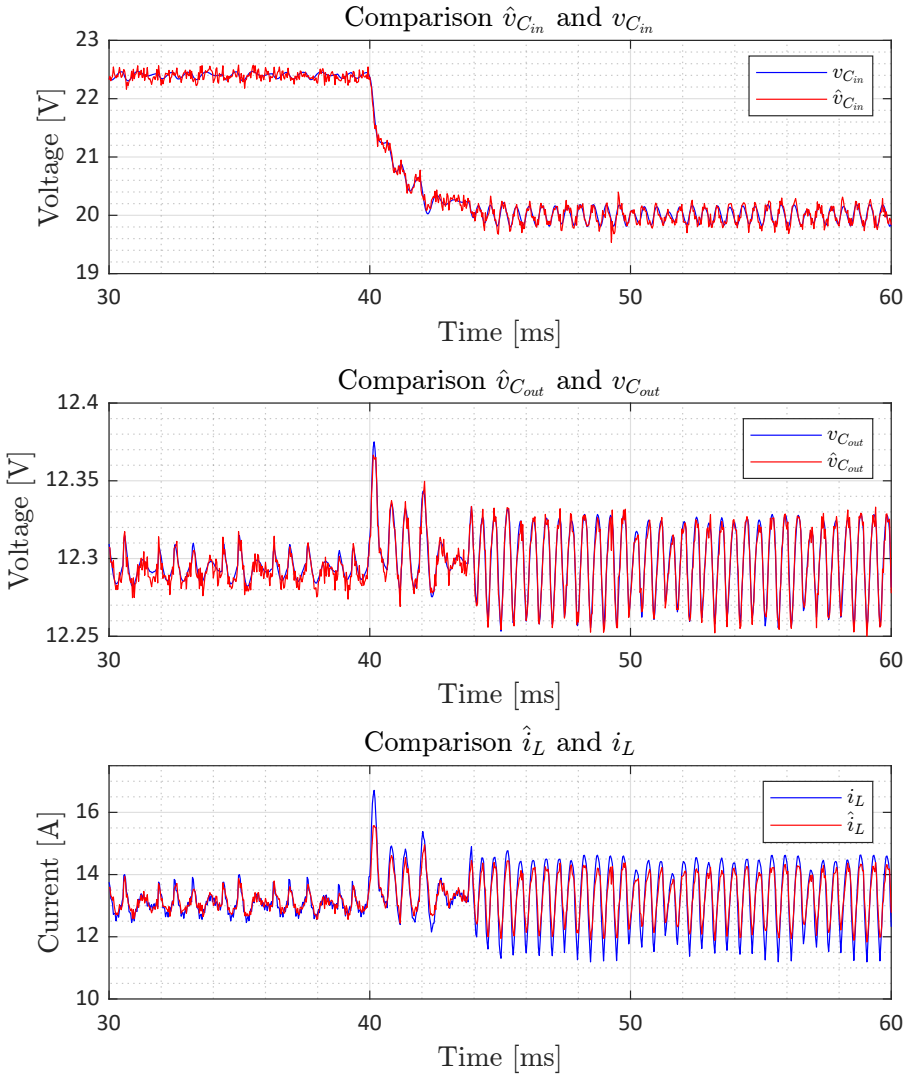


Figure 8.6: Simulation in working point C (buck) with cascade EKF

Chapter 9

Experimental validation

After designing, analyzing and comparing the different control techniques described in the previous chapters, the final stage of the project is the experimental validation. Testing the algorithms on a real plant allows not only to verify the time performances results obtained in simulation, but also to understand the complexity and the additional difficulties related to a real-time implementation. Following an Hardware-In-the-Loop approach, a prototype of the NIBB is tested, connected to a microcontroller unit that applies suitable command inputs, depending on the control law technique loaded.

In this chapter, after a detailed description of the experimental setup, all the experimental results are shown and described, outlining the practical problems encountered and the proposed solutions.

9.1 Experimental setup

The experimental tests are performed on an innovative NIBB prototype, designed in order to comprehend power stage, measurement stage, supplies and communication interfaces. The prototype is highly efficient, reaching a 95% of efficiency in almost all the working points. The numerical values of the plant components are exactly equal to the ones previously introduced in chapter 2, where the model of the converter is described: all the components are chosen in order to reach high performances with a switching frequency of 150 kHz

The prototype is then controlled thanks to a TMS320F2837xD Dual-Core microcontroller (**MCU**), mounted on a suitable docking station that provides the connections between the microcontroller and the converter. The microcontroller is characterized by a maximum clock frequency of 200 MHz, a IEEE 754 single-precision Floating-Point Unit (**FPU**) and a Trigonometric Math Unit (**TMU**). The load of the control logic on the microcontroller is accomplished through CAN communication.

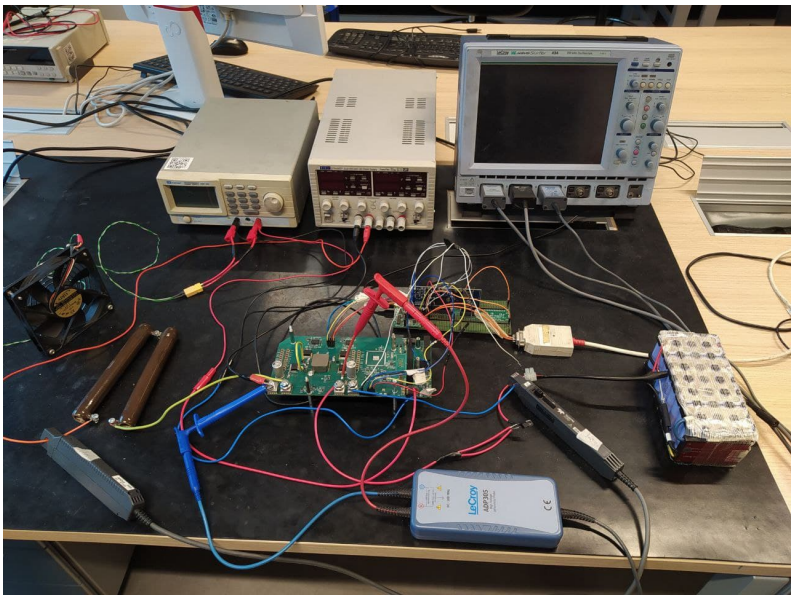


Figure 9.1: Test bench for experimental validation

Two different power generators are used: one that provides a suitable 12 V voltage, in order to supply all the active stages of the converter, and another one that is connected to the input stage of the converter, in series with a 2Ω resistor, for simulating the external input voltage provided by the TEG. Moreover, the output stage of the prototype is connected to a battery pack with a nominal voltage between 14 V and 15 V. All the components are clearly shown in figure 9.2.

A tester and an oscilloscope are used in order to measure the quantities of interest. The oscilloscope is equipped with two current probes, one differential voltage probe and a standard probe, which allow to measure the voltages and the currents provided by the measurement stage of the prototype. A fan is employed for containing the heating of the input resistors.

9.2 General settings

In order to load the control logic on the MCU, the Simulink Embedded Coder Support Package for Texas Instruments C2000 processors is exploited. This package provides suitable Simulink blocks for code generation for embedded processors of the Texas Instruments family.

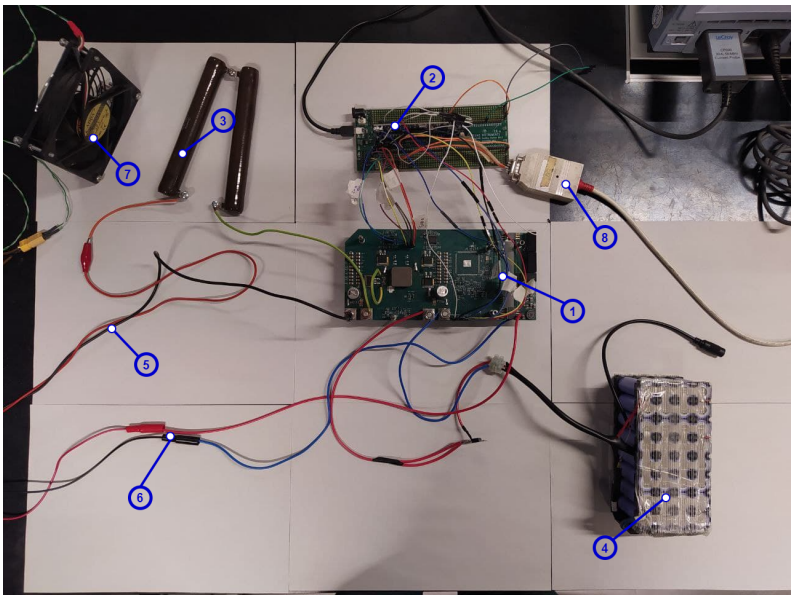


Figure 9.2: 1) NIBB prototype 2) Microcontroller with docking station 3) Input resistor 4) Battery pack 5) Input voltage generator 6) NIBB power supply 7) Fan 8) CAN cable

9.2.1 PWM setting

The first functionality to be set is the PWM duties generation. As described in the previous chapters, the duties of the two switching legs are the command inputs provided to the plant. The duties generation, following the dual carrier approach, can be reached correctly setting the ePWM blocks provided by the Support Package. The ePWM blocks configure the event manager of the MCU to generate ePWM wave-forms: two ePWM blocks are exploited, one related to the buck leg and one related to the boost leg. Each ePWM unit of the MCU is equipped with two channels, of which the A channel is exploited for the high side switch of the leg and the B channel for the low side switch of the leg.

The first ePWM block is set with a switching frequency of 150 kHz, generating interrupts for the execution of the control algorithm. The second ePWM is linked and synchronized with the first one: this is important in order to have the same period of switching and in order to avoid unwanted phase shifts. The second ePWM block generates interrupts for the Start of Conversion (**SOC**) of the ADC modules used. Both the PWM modules work in up-down counting mode. The duties of the wave-forms are set through external inputs, that are the commands given by the controller or by the user through keyboard, depending on the purpose of the test. Also the two wave-forms are shifted of 180 deg, imposing the same settings of clear and set for the two ePWMs, but giving complementary duty cycles.

A series of testes were carried out in order to define the dead-band region of the

PWMs. The dead-band region (or dead-time) is fundamental in order to avoid to short circuit one leg of the converter, causing damages to the switches. However, the dead-band region has not to be too large, otherwise a large range of duty cycles cannot be produced. Initially, the dead time was set as 500 ns, as can be seen in figure 9.3.

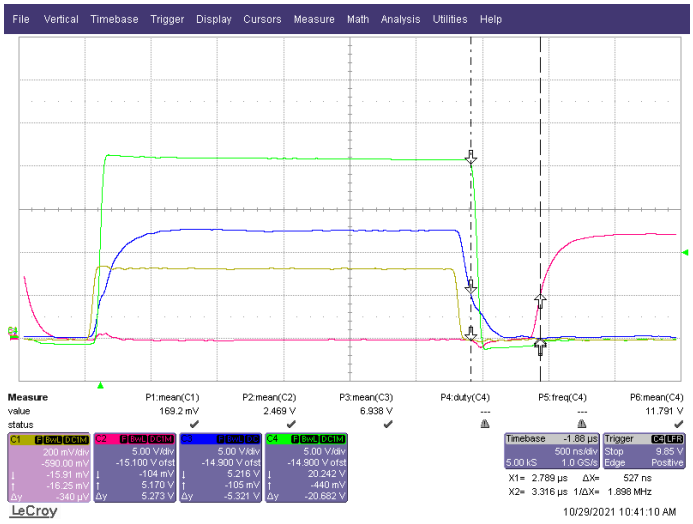


Figure 9.3: Buck leg PWM wave-forms with 500 ns dead-time

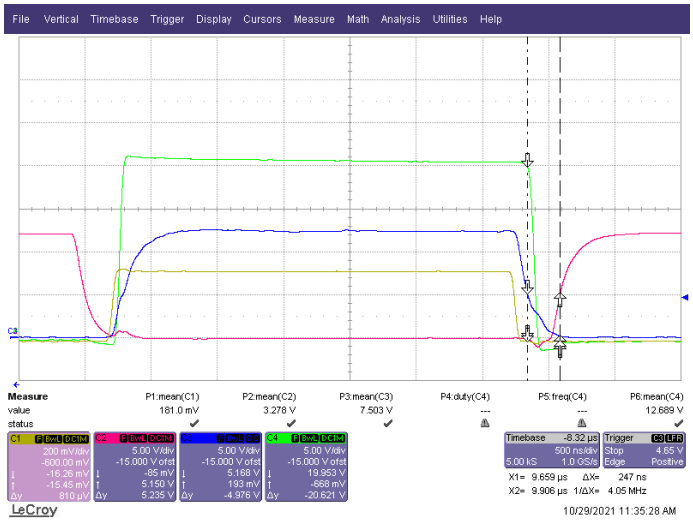


Figure 9.4: Buck leg PWM wave-forms with 250 ns dead-time

Figures 9.3 and 9.4 represent:

- the gate voltage of the buck leg high-side switch (blue);
- the gate voltage of the buck leg low-side switch (pink);
- the driver output (yellow);
- the switching point (green).

A 500 ns dead-time leads to a 1 μs time lost, that, considering a switching frequency of 150 kHz (equivalent to a switching period of 6.67 μs), corresponds to the 15% of the switching period itself. However, figure 9.3 shows that a margin of reduction is possible: the difference between the switching point and the wave-form of the high side switch allows to define an inferior limit of about 90 ns. In order to be enough conservative, a second test is performed with a dead-time of 250 ns. The results, observable in figure 9.4, proof the effectiveness of this choice.

9.2.2 ADC setting

The second element to be configured is the ADC of the MCU: the TMS320F2837xD allows to sample and convert external signals to digit, through four different ADC modules that can work in parallel. The ADCs can be set in single or differential coupling. The single measure has a maximum resolution of 12 bits, while the differential measure allows to reach a resolution of 16 bits, with noise reduction. Because four different measures are available ($v_{C_{in}}$, $v_{C_{out}}$, i_{in}, i_{out}), the four ADC modules are employed. All the modules have the same SOC, in order to start the conversion at the same time, while the End of Conversion (**EOC**) can be triggered indifferently by one of the module (in this case, module D). The SOC trigger is produced by the second PWM and it differs depending on the value of duty cycles and on the working mode of the converter. This is fundamental in order to avoid to sample the noise produced by the change of state of the switches. Figure 9.5 shows where the sampling is desired. In figure 9.5 are represented:

- the wave-form related to the high-side switch of the buck leg (blue);
- the input capacitor voltage $v_{C_{in}}$ (pink);
- the sampling position (green).

In order to set correctly the sampling point, different tests were carried out, observing the behaviour of the wave-forms of the two couples of switches. The position of the sampling point is defined through a second compare value of the second PWM.

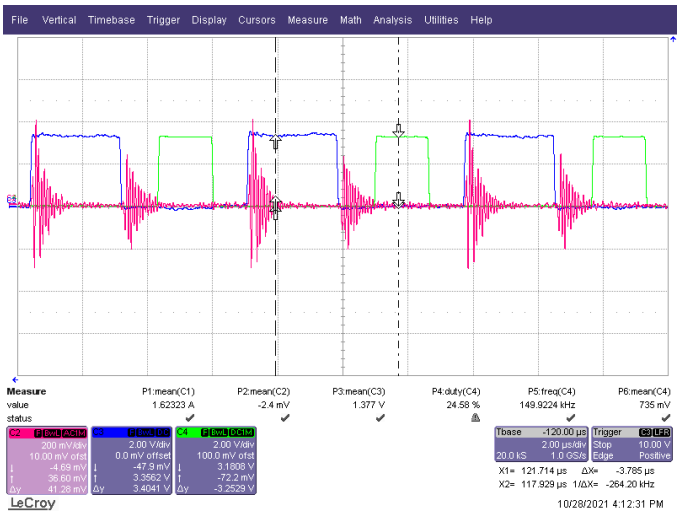


Figure 9.5: Desired ADC sampling point

The compare is the percentage value that allows to define a certain duty cycle: in this case, the PWM is configured for generating a SOC trigger every time the defined compare value is reached, counting from down to up. This compare value is different from the one that defines the duty cycle of the boost leg and it is set as follows:

- in buck mode, if the duty cycle d_a is greater than 60%, the compare value is set to 2%, otherwise the value is set to 40%;
- in boost mode, if the duty cycle d_b is less than or equal to 40%, the compare value is set to 2%, otherwise the value is set to 40%;

These settings avoid to acquire the switching noise, obtaining an accurate measure of the signal of interest. Also, in order to reduce the noises provided by the measurement stage, a window of acquisition of 1 μs is set for all the modules.

Then, through different data acquisitions, the characterization of the ADC stages was performed. The value given by the ADC blocks is not the measure of the actual quantity of interest, but the digital value corresponding to the voltage at the pin of the converter measurement stage. The first operation to perform is so the conversion from the digital value (X_{digit}) to the voltage value (X_{volt}) given by the measurement stage. This can be easily obtained considering the working range of the ADC module and its resolution. Taking into account that the working range is $[-3V\ 3V]$ and that the ADC is working in differential coupling, with a 16 bit resolution, the voltage value is obtained as follow:

$$X_{volt} = 2 \cdot \left(\frac{3}{2^{16} - 1} \cdot X_{digit} - 1.5 \right) \tag{9.1}$$

Relationship (9.1) is valid for all the ADC modules used. Then, the voltage value sampled by the ADC module from the measurement stage has to be converted to the actual value of the quantity of interest. In order to do this, a series of measures in different points were collected, comparing the voltage values sampled by the ADC stage with the actual values of the quantity of interest, observed with the oscilloscope probes. The data collected allow to construct a regression line for each measured quantity (X), defining a gain (a_X) and an offset (b_X) value for accomplish the conversion. Table 9.1 shows the gains and the offset applied, that are computed building the regression line of a series of experimental data.

$$X = a_X \cdot X_{volt} + b_X \tag{9.2}$$

	i_{in}	i_{out}	$v_{C_{in}}$	$v_{C_{out}}$
a_x	16.5029	16.6133	20.3024	6.6478
b_x	0.0213	-0.0083	0.0244	0.1053

Table 9.1: Gains and offsets related to the measured quantities $v_{C_{in}}$, $v_{C_{out}}$, i_{in}, i_{out}

In figures 9.6 and 9.7 it is possible to observe the regression line construction for the measured quantities $v_{C_{in}}$, $v_{C_{out}}$, i_{in}, i_{out}. Also, each figure is followed by a table that resumes the employed collected data. The data are reported as the mean values of the analyzed quantities.

Other tests are carried out in order to analyze and verify the accuracy of the values of gain and offset computed. Three random sample measures are provided for each quantity of interest; the results are all collected in tables 9.4 and 9.5. The analysis shows that the accuracy improves by increasing the value of the measured quantity. This is related to the fact that, with a 16 bit resolution, the ADC is not able to distinguish very well low values of voltages: for example, if the input current is between 0 and 1, the ADC associates the same value to all the currents in the range.

In tables 9.4 and 9.5, the "bar" value ($\overline{x}$) is the voltage measure sampled by the ADC, the "hat" value ($\hat{x}$) is, instead, the value obtained by applying the corrections evaluated with the regression line. The quantities without mark (x) are measured with the oscilloscope.

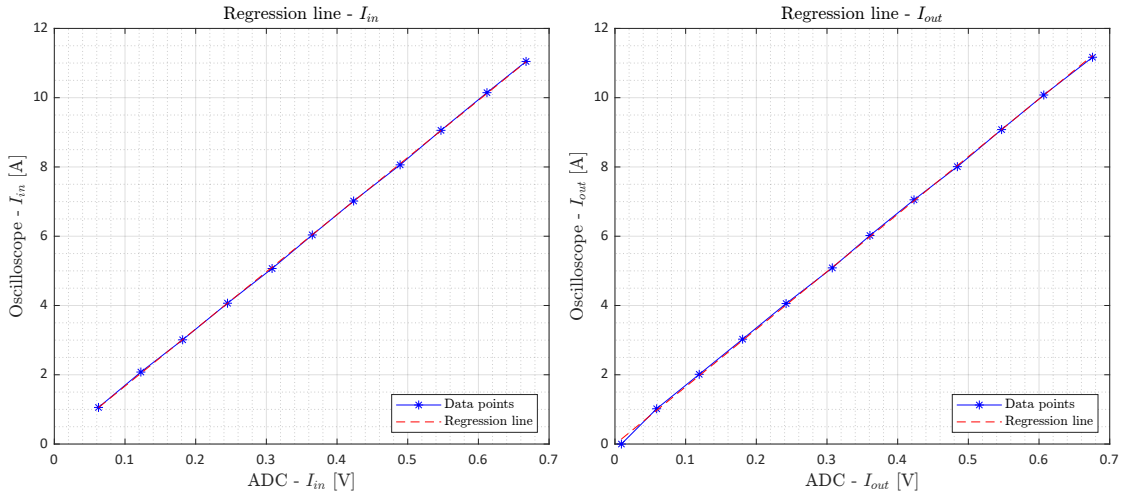


Figure 9.6: Regression lines for the input current i_{in} and the output current i_{out}

i_{in} [A]	$\bar{i}_{in}$ [V]	i_{out} [A]	$\bar{i}_{out}$ [V]
1.0561	0.06251	0	0.00913
2.0782	0.12244	1.0186	0.05900
3.0096	0.18156	2.0130	0.11923
4.0657	0.24504	3.0278	0.18062
5.0679	0.30810	4.0559	0.24211
6.0369	0.36499	5.0867	0.30748
7.0169	0.42346	6.0214	0.36077
8.0608	0.48926	7.0515	0.42312
9.0575	0.54704	8.0089	0.48447
10.1450	0.61213	9.0795	0.54702
11.0464	0.66743	10.0780	0.60672
12.1476	0.73480	11.1679	0.67570

Table 9.2: i_{in} and i_{out} data points used for the regression lines

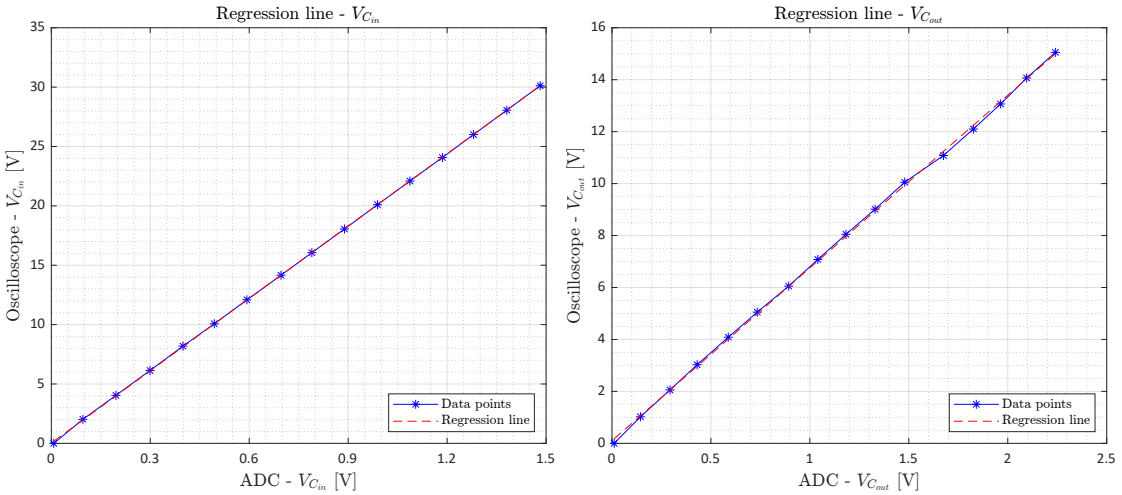


Figure 9.7: Regression lines for the input capacitor voltage $v_{C_{in}}$ and the output capacitor voltage $v_{C_{out}}$

$v_{C_{in}}$ [V]	$\overline{v}_{C_{in}}$ [V]	$v_{C_{out}}$ [V]	$\overline{v}_{C_{out}}$ [V]
0	0.00734	0	0.00913
2.0243	0.09566	1.0233	0.01109
4.0393	0.19583	2.0570	0.14429
6.1364	0.29906	3.0275	0.29331
8.1897	0.39920	4.0869	0.43052
10.0732	0.49405	5.0505	0.58779
12.0893	0.59296	6.0545	0.73421
14.1596	0.69644	7.0789	1.04000
16.0573	0.78992	8.0480	1.18254
18.0509	0.88879	9.0151	1.32964
20.0994	0.98886	10.0542	1.47937
22.0903	1.08716	11.0818	1.67519
24.0690	1.18566	12.1016	1.82642
26.0055	1.28016	13.0664	1.96393
28.0590	1.38092	14.0667	2.09545
30.1164	1.48210	15.0506	2.24068

Table 9.3: $v_{C_{in}}$ and $v_{C_{out}}$ data points used for the regression lines

i_{in} [A]	$\hat{i}_{in}$ [A]	e_r	i_{out} [A]	$\hat{i}_{out}$ [A]	e_r
1.4520	1.4311	1.4360%	1.6642	1.5672	5.8280%
5.0631	5.0154	0.9429%	5.8524	5.8243	0.4807%
9.0457	9.0625	0.1863%	10.2241	10.2467	0.2206%

Table 9.4: Accuracy analysis of ADC conversion of input and output currents

$v_{C_{in}}$ [V]	$\hat{v}_{C_{in}}$ [V]	e_r	$v_{C_{out}}$ [V]	$\hat{v}_{C_{out}}$ [V]	e_r
3.3906	3.4031	0.3698%	1.4624	1.6154	10.4595%
15.2117	15.2810	0.4558%	5.7403	5.7850	0.7787%
30.6930	30.8235	0.4251%	12.4899	12.4150	0.5996%

Table 9.5: Accuracy analysis of ADC conversion of input and output capacitor voltages

9.2.3 Other settings

The project is divided in two main tasks: the ADC sampling and the control law. The control law is defined as a preemptable task with highest priority, while the sampling of the ADC is a non-preemptable task, ensuring that the ADC sampling is always performed. Two General Purpose Input/Output (**GPIO**) are exploited, in order to verify the execution of both ADC sampling and control law. The code is auto-generated and loaded on the board through a custom linker application, which communicates with the controller using CAN protocols.

9.3 Kalman filter implementation

Before the implementation of the control laws, the Kalman filter has to be deployed. The EKF is fundamental for the control law application, not only because the filter reduces the effect of the noise on the measurements, but also because it has to compute an estimate of the missing state i_L, that is not provided by the measurement stage of the converter prototype.

Initially, the EKF algorithm was implemented as written for the simulation of the single EKF, exploiting the functionality of the second CPU core of the MCU. The first CPU is devoted to the ADC sampling and to the command reception and application. The measured quantities, sampled by the ADC, and the duty cycles, provided manually by the user, are transmitted through Inter Processors Communication (**IPC**) modules to the second CPU, where the EKF algorithm is loaded. In order to reach a good enough estimation, the EKF was slightly re-tuned with respect to the simulation value:

- the initial state is adapted to the parameters of the real plant, that differs from the model in some components, such as the input resistor, that is equal to 2Ω, and the voltage of the battery, that is in between 14 V and 15 V;

$$\hat{x}(1|0) = \begin{bmatrix} 0 & 14 & 0 & 0 & 14 & 2 & 22.5e-3 \end{bmatrix}^T$$

- the process noise variance matrix V_1 is modified, in order to takes into account for the uncertainty introduced by the real plant. A modification is provided also for the measure noise variance matrix V_2.

$$V_1 = \text{diag}(1{,}1{,}500{,}0{,}1{,}0{,}1) \qquad V_2 = \text{diag}(0.001{,}0.001{,}0.001{,}0.001)$$

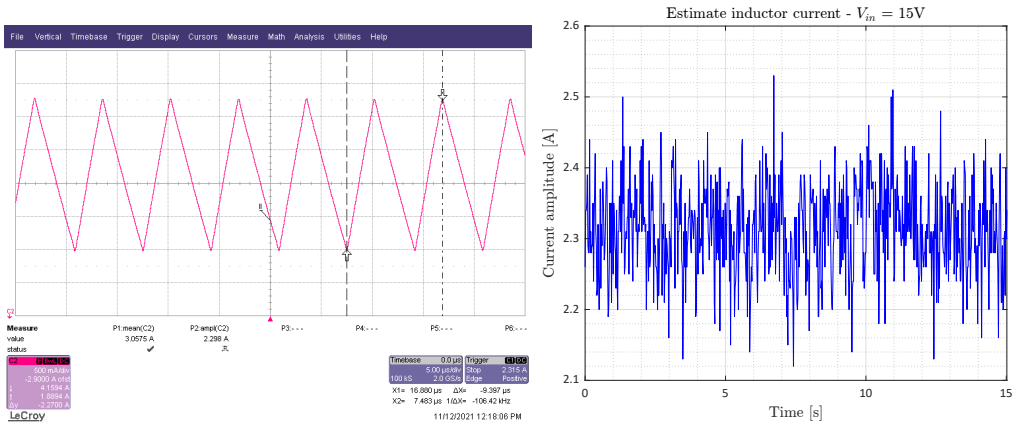


Figure 9.8: Comparison between the measured and the estimate value of the inductor current - Boost mode ($v_{C_{in}} = 9.04\,\text{V}$, $v_{C_{out}} = 14.84$, $L = 10\,\mu\text{H}$)

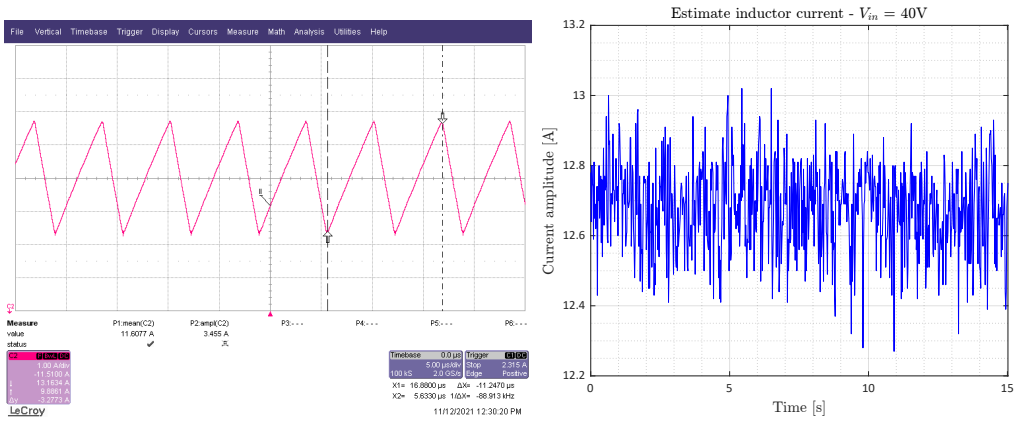


Figure 9.9: Comparison between the measured and the estimate value of the inductor current - Buck mode ($v_{C_{in}} = 23.35\,\text{V}$, $v_{C_{out}} = 15.92$, $L = 10\,\mu\text{H}$)

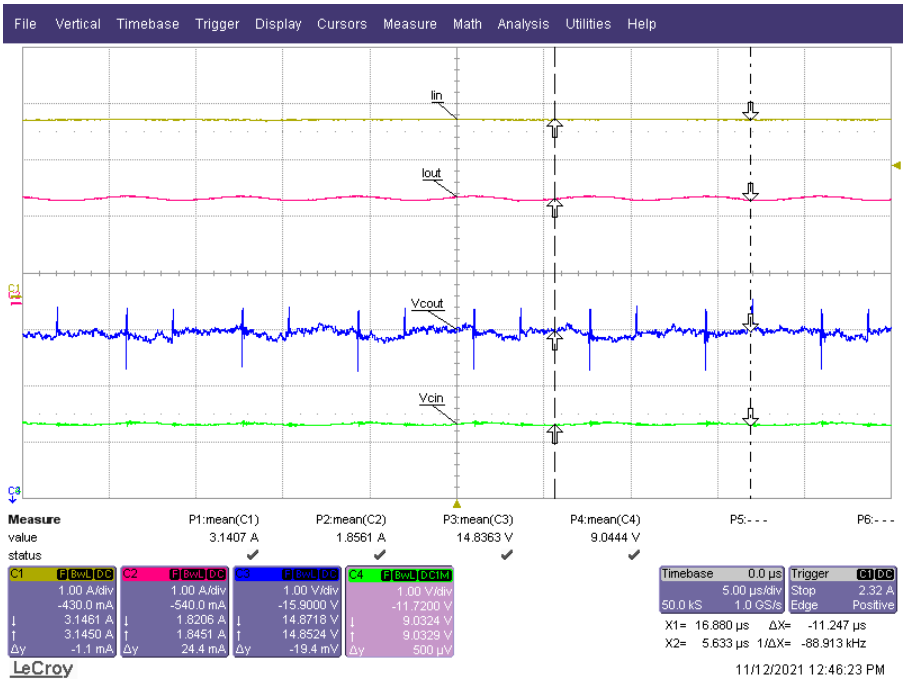


Figure 9.10: Measured states of the system - Boost mode ($v_{in} = 15$)

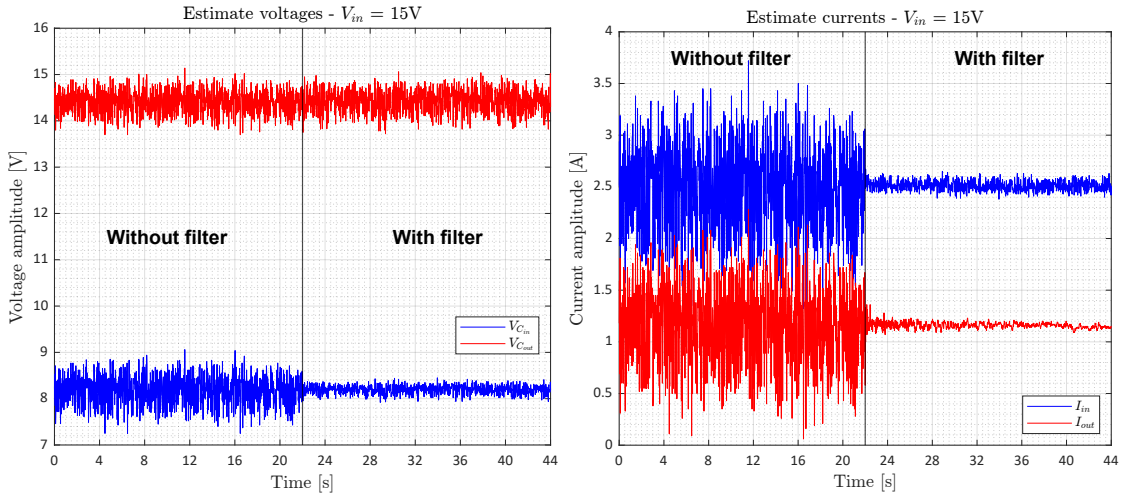


Figure 9.11: Comparison between the measures without filter and with the filter - Boost mode ($v_{in} = 15$)

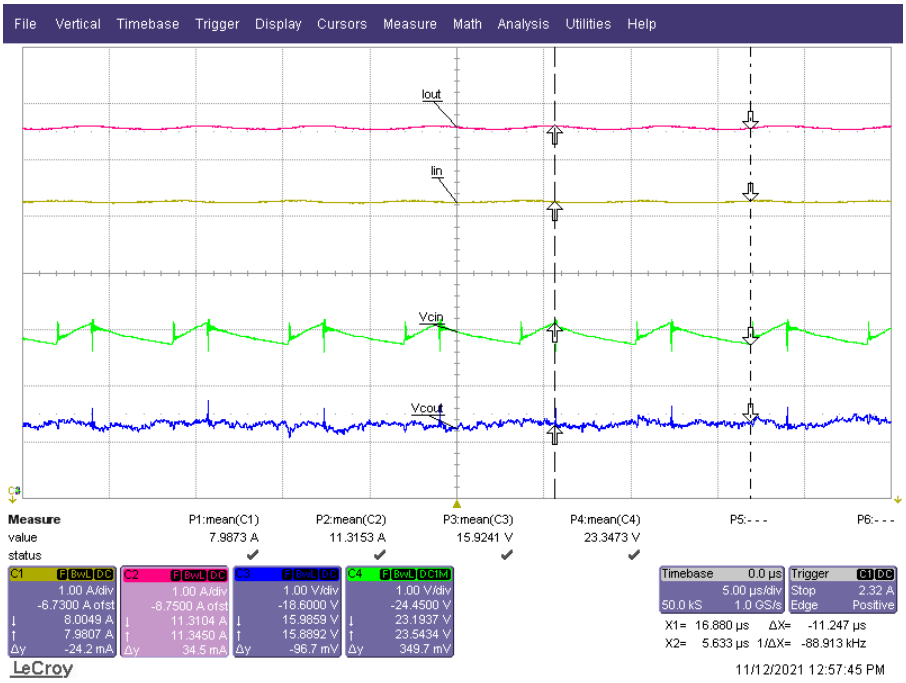


Figure 9.12: Measured states of the system - Buck mode ($v_{in} = 40$)

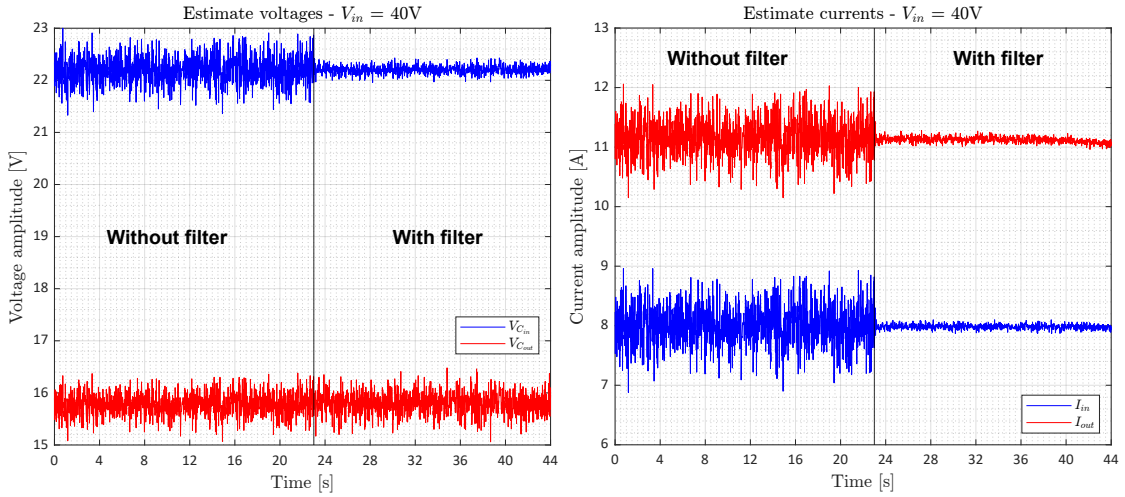


Figure 9.13: Comparison between the measures without filter and with the filter - Buck mode ($v_{in} = 40$)

However, as it is, the EKF is not able to work at high frequencies: the maximum working frequency at which the EKF can operate on the MCU is 4 kHz. A so low frequency does not ensure, even in simulation, the convergence of the estimate to the true value of measure.

Figures 9.8 and 9.9 compare the estimate of the inductor current with the actual measure: the comparison shows that the estimate has an important offset with respect to the true value, that is caused by the working frequency of the filter. As a matter of facts, the minimum working frequency that ensures a good estimate of the inductor current is 30 kHz, that is about ten times greater than the one possible with the MCU implementation. However, the EKF seems to reduce effectively the noise on the available measures, as shown in figures 9.11 and 9.13.

In order to try to reduce the computational complexity of the filter, the algorithm was structured in a different way, employing the conversion from floating-point data type to fixed-point data type. Because the matrix inversion is not possible in fixed-data type computation, the algorithm is divided into two different sub-tasks: the computation of the Kalman gain and the estimate production.

$$K_0(k) = P(k)\hat{C}(k)^T[\hat{C}(k)P(k)\hat{C}(k)^T + V_2]^{-1} \tag{9.3}$$

The computation of the Kalman gain K_0 requires the inversion of a 4-by-4 matrix (see (9.3)), that is not possible using fixed-point data types. For this reason, the Kalman gain computation is divided from the estimation itself and computed in floating-point data type. The data type chosen is the Matlab 'single-precision' data type, that corresponds to a 32 bit floating-point data type. The estimate, instead, is produced using a fixed-data type with 16 bits for the word, 1 bit for the sign and different types of fraction length, depending on the variable taken into account. This type of conversion produces interesting results if compared with the floating-point implementation: the comparison is shown through the RMSE value computation, considering as reference for the fixed-point simulation the floating-point model of the EKF itself.

$var \backslash W.P.$	A	B	C	D
$v_{C_{in}}$	0.05808	0.05682	0.05735	0.05796
$v_{C_{out}}$	0.01327	0.02256	0.10810	0.21770
i_L	1.93400	0.79170	1.10200	0.68860
i_{in}	0.13890	0.13890	0.14120	0.14430
i_{out}	0.58800	0.57650	0.58220	0.58770

Table 9.6: Values of RMSE of the EKF in fixed-point for four different working points

As outlined in table 9.6, the results obtained with a fixed-point data type are quite close to the results that can be obtained with a model defined in floating-point data type. However, the fixed-point algorithm requires additional checks and conditions in order to avoid the bit growth while performing mathematical operation. This leads to an added complexity that slows down the execution of the filter, reaching a maximum working frequency of about 2 kHz.

Also the usage of the Control Law Accelerator (**CLA**) was taken into account for reducing the execution time related to the matrix inversion. The basic idea was to implement the matrix inversion task into the CLA of the second CPU. However, the computation of the Kalman gain requires an amount of memory not available in the CLA data memory, making this kind of implementation unfeasible.

9.4 Control implementation

The implementation of the control algorithm was performed starting from the design results obtained in simulation. For each controller, the command signal is translated in suitable duty cycles for the buck and the boost leg, using the dual-carrier approach. Also, the control law is coupled with the possibility of defining a set point with an open-loop control: the set point can be imposed providing suitable duty cycles with the keyboard, then the closed-loop control can be activated.

9.4.1 PI control

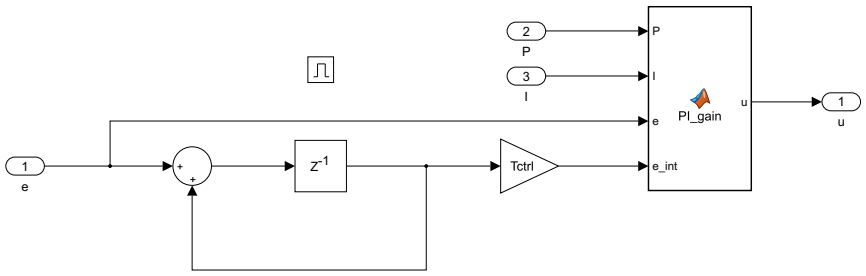
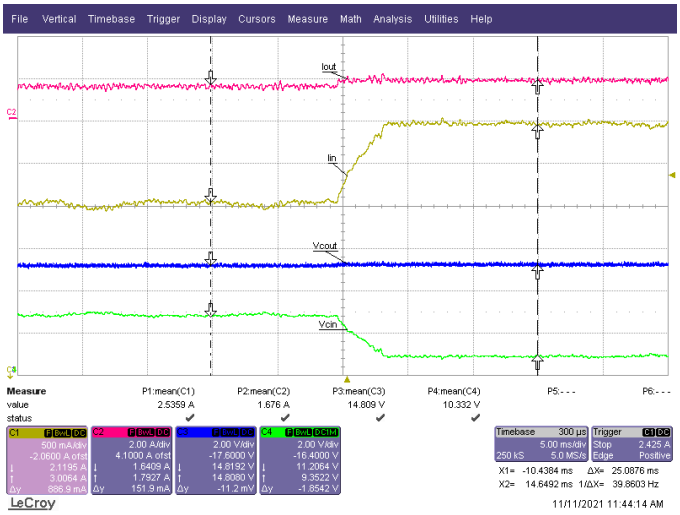


Figure 9.14: Scheme of the PI control algorithm for code generation

As described in chapter 3, the PI controller is a very effective control algorithm for the most problem cases. In this case, the gains are defined through the function `PI_gain`, as shown in figure 9.14, allowing to tune the parameters with the keyboard. In this case, the values for P and I chosen with the design procedure give good enough performances, close to the ones observed in simulation. Two tests were performed, one in boost mode and one in buck mode, with the controller working at 30 kHz.



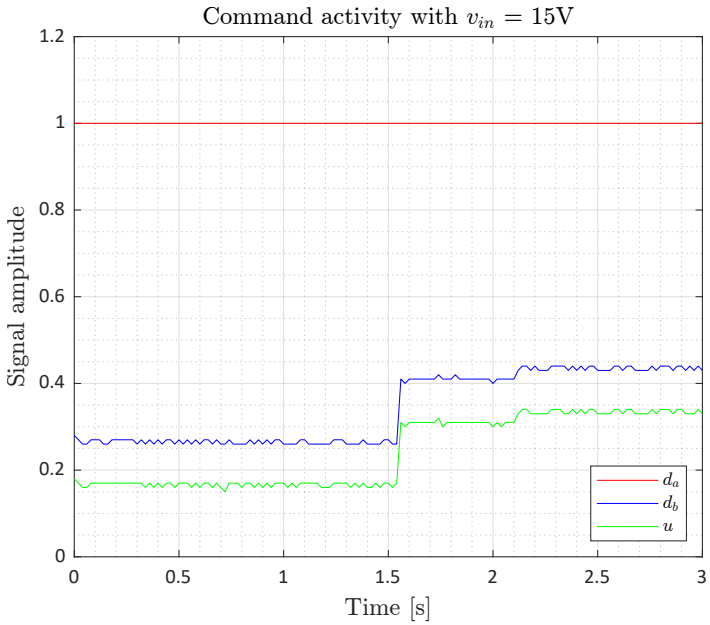


Figure 9.15: Test of PI controller with $v_{in} = 15\,\mathrm{V}$ (boost). On top, the control variable is observed together with the other measured quantities. Bottom, the command activity is represented

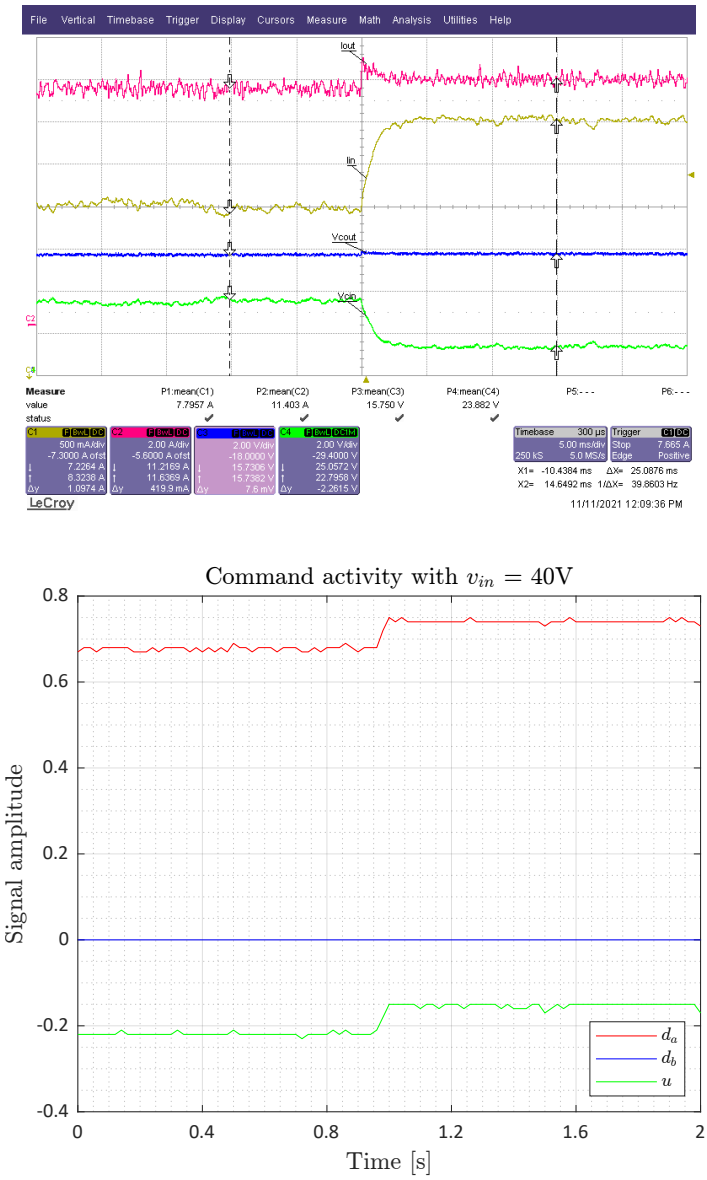


Figure 9.16: Test of PI controller with $v_{in} = 40\,\mathrm{V}$ (buck). On top, the control variable is observed together with the other measured quantities. Bottom, the command activity is represented

As for the previous chapters, the performances are evaluated through the function `time_performance` and then listed in the table below.

v_{in} [V]	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
15	2.69	3.48	3.49
40	1.02	1.66	1.09

Table 9.7: Values of overshoot, rise time and settling time for $v_{in} = 15\,\mathrm{V}$ and $v_{in} = 40\,\mathrm{V}$ for PI controller in hardware implementation

As shown from figures 9.15, 9.16 and table 9.7 the performances of the hardware implementation are very close to the ones observed in simulation: the differences are caused by the offset and the noise of the measures (that are not filtered) and by the values of the battery voltage and the input resistor, that are not equal to the ones exploited in simulation.

9.4.2 H-infinity control

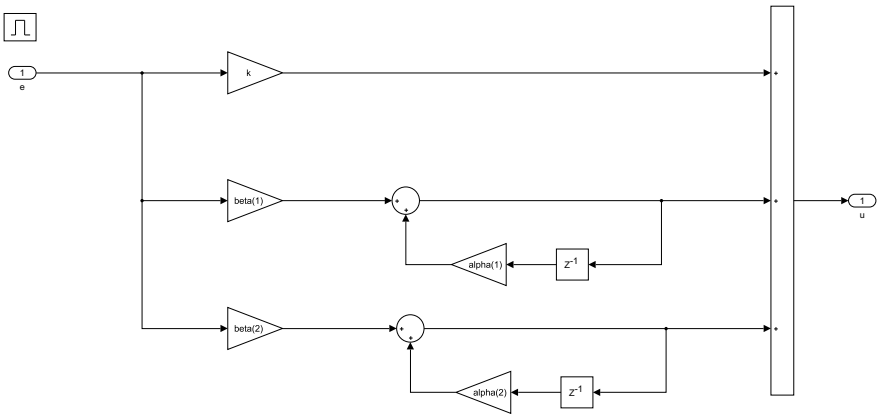
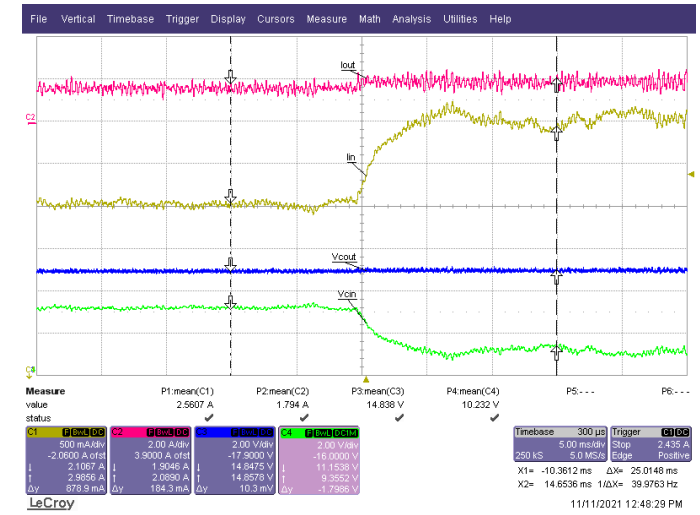


Figure 9.17: Scheme of the LIN control algorithm for code generation

For the H-infinity controller implementation, the transfer function employed is the same provided during the design in DT, using a pre-multiplication architecture that allows to reduce the multiplication errors provided by the gains. The transfer function exploited is recalled in equation (9.4).

$$C(z) = \frac{0.021479(z^2 - 1.847z + 0.8947)}{(z-1)(z-0.6613)} \tag{9.4}$$

Two tests were performed, one in boost mode and one in buck mode, with the controller working at 30 kHz of frequency.



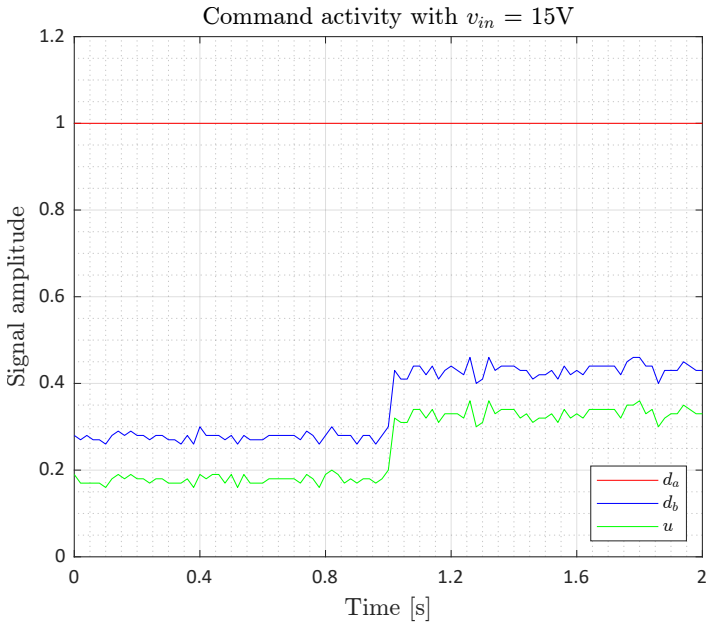


Figure 9.18: Test of H-infinity controller with $v_{in} = 15\,\mathrm{V}$ (boost). On top, the control variable is observed together with the other measured quantities. Bottom, the command activity is represented

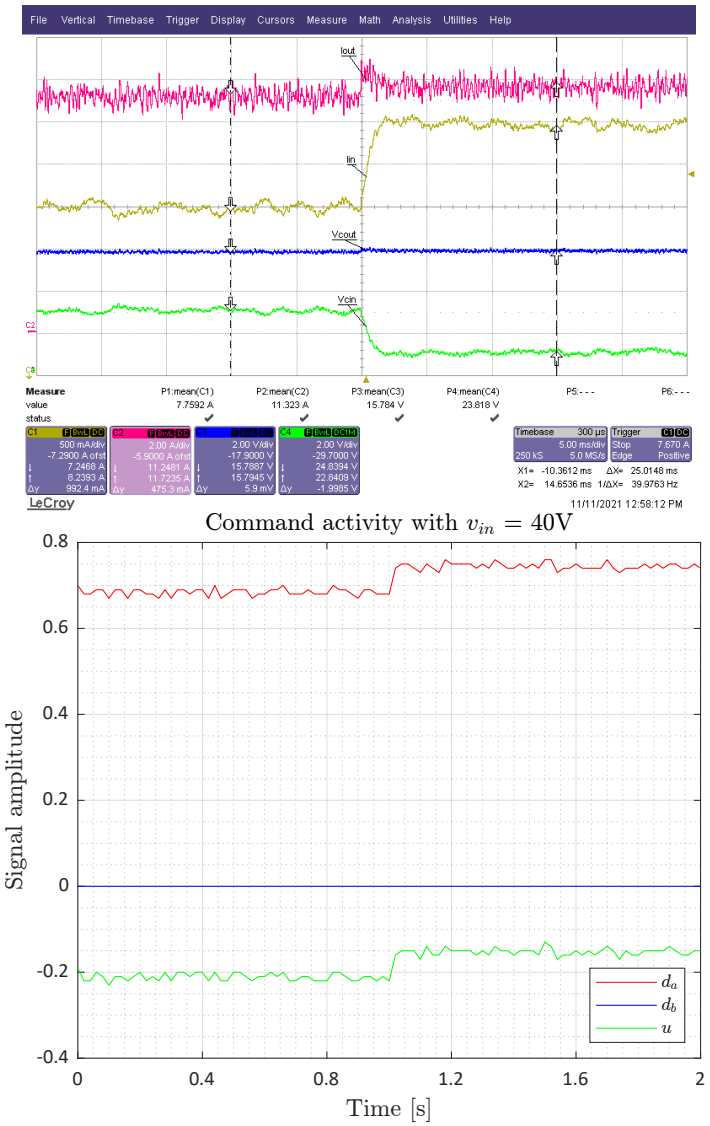


Figure 9.19: Test of H-infinity controller with $v_{in} = 40\,\mathrm{V}$ (buck). On top, the control variable is observed together with the other measured quantities. Bottom, the command activity is represented

As for the previous chapters, the performances are evaluated through the function `time_performance` and then listed in the table below.

v_{in} [V]	$\hat{s}_{\%}$	t_r [ms]	$t_{s,3\%}$ [ms]
15	6.23	3.96	15.76
40	1.51	1.08	0.78

Table 9.8: Values of overshoot, rise time and settling time for $v_{in} = 15\,\mathrm{V}$ and $v_{in} = 40\,\mathrm{V}$ for H-infinity controller in hardware implementation

As shown from figures 9.18, 9.19 and table 9.8 the performances of the hardware implementation are enough close to the ones observed in simulation: again, the difference is related to the fact that the battery voltage and the input resistor are not equal to the values used for simulation and, also, to the noise and the offsets that affect the measures. The variation of the output voltage and of the input resistor values is more important for this type of controller, that is tailored for the working points of the simulation plant.

9.4.3 Linear quadratic regulator control

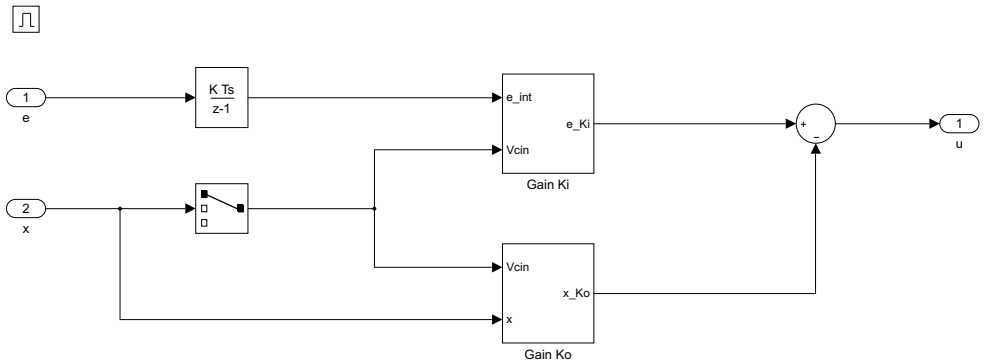


Figure 9.20: Scheme of the LQR control algorithm for code generation

Even if the EKF cannot be used, due to its low working frequency, a test with the LQR was tried. Because the gains change depending on the working mode, as outlined in chapter 5, the controller algorithm is completed with a switch block, that defines the values of the direct gain and of the state feedback gain depending on the working mode.

Moreover, because the EKF is not available, a simple observer is defined in order to give a rough estimate of the inductor current. This observer provides an approximated value of the inductor current using the equations computed with the big signal analysis of the converter, so that:

- $i_L = 0$ if the inductor is short circuited, with $d_b = 1$;
- $i_L = i_{out}/(1 - d_b)$ otherwise.

The LQR is implemented using the same parameters computed in chapter 5.

$$\begin{aligned} K_{boost}^{DT} &= \begin{bmatrix} -0.1708 & -0.0093 & 0.0328 & 756.7962 \end{bmatrix} \\ K_{buck}^{DT} &= \begin{bmatrix} -0.1783 & -0.0182 & 0.0167 & 1.6176e3 \end{bmatrix} \end{aligned} \tag{9.5}$$

However, due to the absence of the EKF, that is fundamental not only for the estimation of the inductor current but also for the noise cancellation on the available measures, the LQR is not able to work properly. Using a working frequency of 50 kHz, the control command saturate to the upper limit and the correct control action is not achieved, as shown in figure 9.21. As a matter of facts, also the difference with the input resistor and the battery voltage should be taken into account.

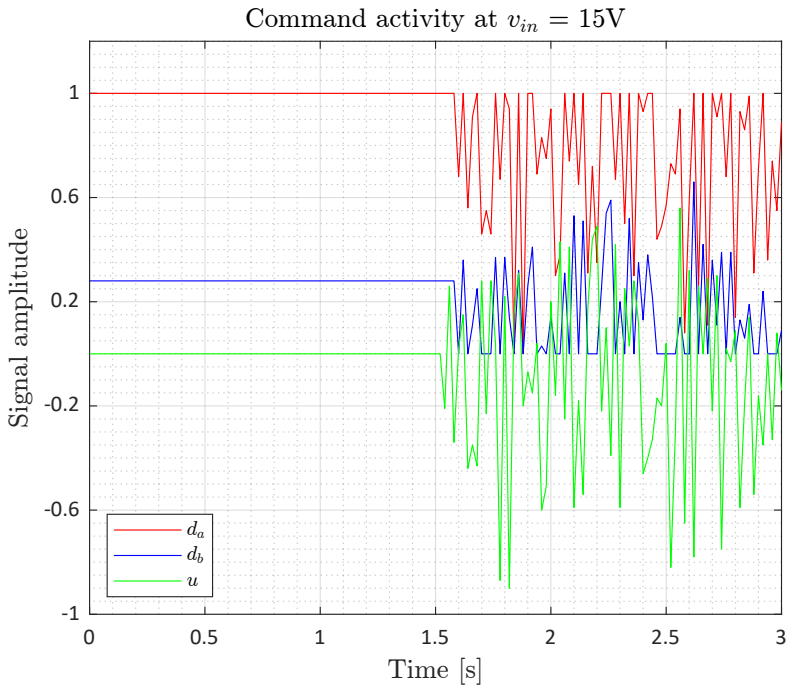


Figure 9.21: Command activity of the LQR with $v_{in} = 15\,\text{V}$ (boost)

A working frequency higher than 50 kHz cannot be reached. However, this is not a problem, because the gains values computed considering a working frequency of 75 kHz allow to obtain, in simulation, the same time performances with the controller running at 50 kHz.

9.4.4 Sliding mode control

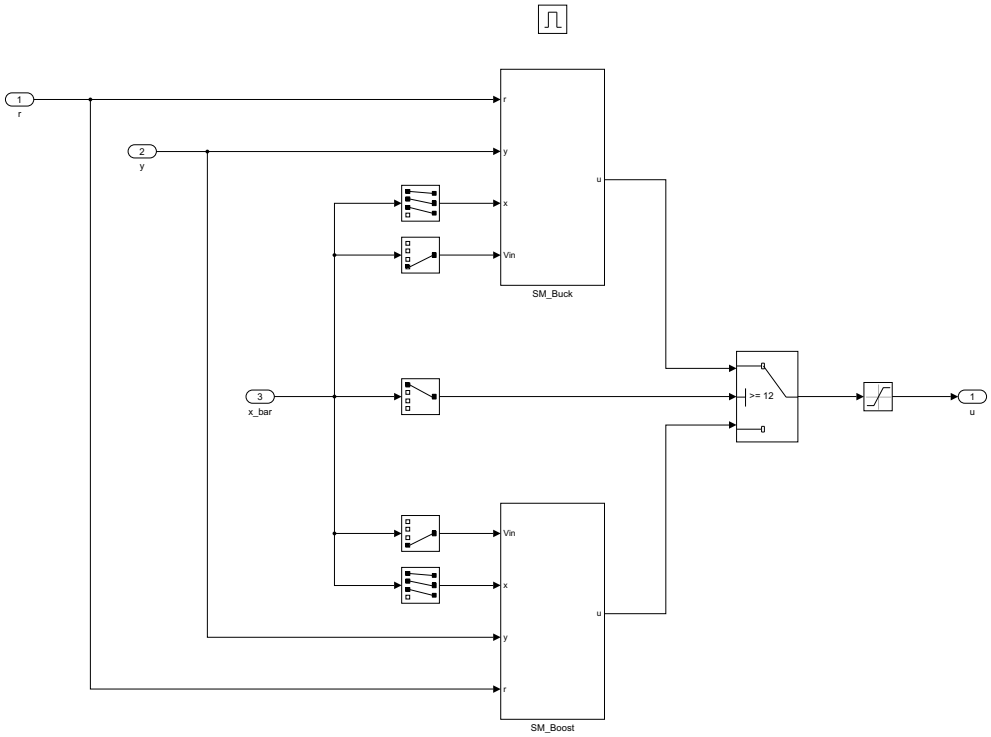


Figure 9.22: Scheme of the SM control algorithm for code generation

Even if the EKF cannot be used, due to its low working frequency, also a test with the SM was tried. Because the sliding surface changes depending on the working mode, as outlined in chapter 6, the controller algorithm is completed with a switch block, that defines the sliding surface function depending on the working mode.

Moreover, because the EKF is not available, a simple observer is defined in order to give a rough estimate of the inductor current and of the input voltage. This observer provides an approximated value of the inductor current, using the equations computed with the big signal analysis, and of the input voltage, exploiting the Kirchhoff voltage law on the input mesh:

- $v_{in} = v_{C_{in}} + i_{in} \cdot R_{in}$, knowing that $R_{in} = 2\Omega$;
- $i_L = 0$ if the inductor is short circuited, with $d_b = 1$;
- $i_L = i_{out}/(1 - d_b)$ otherwise.

The SM is implemented using the same parameters computed in chapter 6 and resumed in table 9.9 for clarity sake.

Parameter	s_1	s_2	s_3	η
Boost value	1e7	3e4	8e8	1e-5
Buck value	1e7	1e3	\	5e-5

Table 9.9: Parameters values for SM control design in DT

Due to the absence of the EKF, that is fundamental not only for the estimation of the inductor current but also for the noise cancellation on the available measures, the SM is not able to work properly. Indeed, using a working frequency of 30 kHz, the control command saturate to the upper limit and the correct control action is not achieved, as shown in figure 9.23. Another factor can be the approximation of the control parameters themselves: some values are too big to be represented correctly, leading to a precision lost that can be problematic for this type of control law. As a matter of facts, also the difference with the input resistor and the battery voltage should be taken into account.

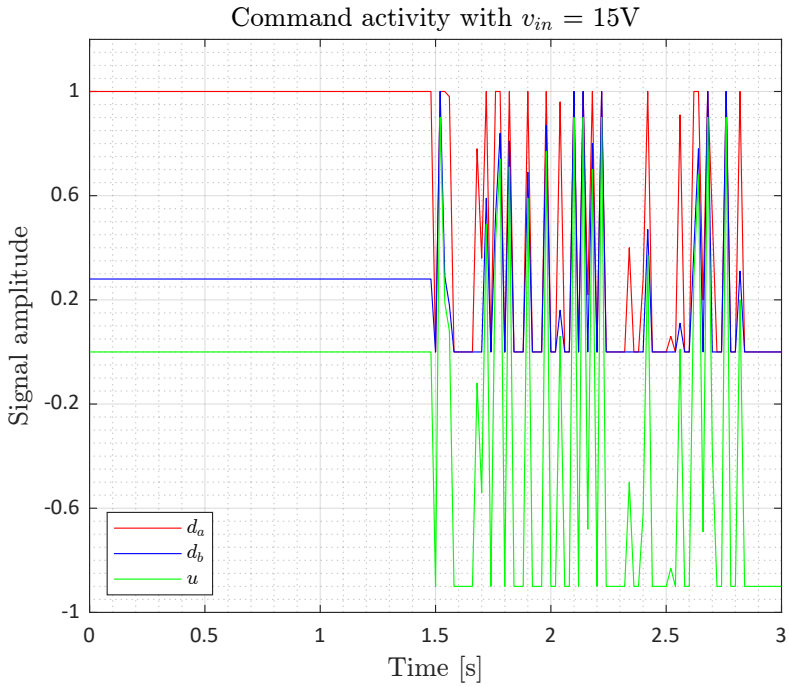


Figure 9.23: Command activity of the SM with $v_{in} = 15\,\mathrm{V}$

9.4.5 Fuzzy logic control

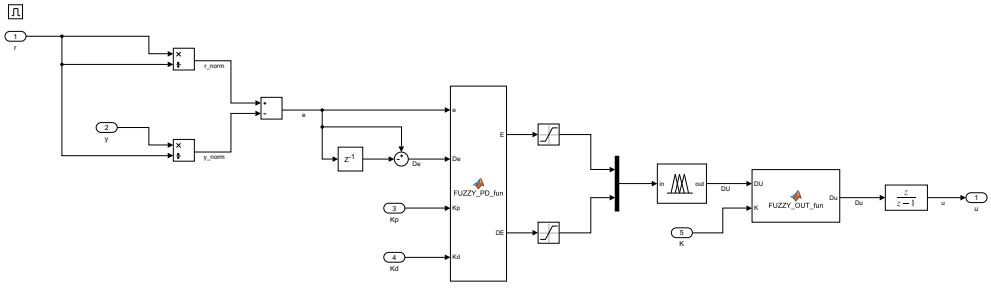


Figure 9.24: Scheme of the Fuzzy logic control algorithm for code generation

The FL controller is the last control algorithm implemented. As described in chapter 7, this controller is based on 49 inference rules, that provide a suitable command depending on the value of the fuzzy input variables. However, these rules introduce a very high number of conditions that the MCU is not able to elaborate. The MCU cannot run the controller law at a suitable frequency, making the algorithm totally unfeasible, as it is.

A possible solution to this problem is the application of a subtractive clustering, as suggested by [43], collecting and normalizing a series of suitable data that have to be used for the clustering procedure. The subtractive clustering, indeed, seems able to drastically reduce the number of rules to apply, exploiting the implicit knowledge inside the 49 expert rules already employed.

Chapter 10

Conclusion

This last chapter concludes the thesis dissertation, summing up the achieved results and the future works that may be carried out for more improvements

10.1 Achieved results

The objective of this project was to find an effective control law, able to control the DC-DC converter with the best performances possible: small rise time, small settling time and overshoot as close as possible to zero. In order to achieve this, different control techniques are analysed, with different types of design and complexity. Starting from the simplest approach, the PI control, more advanced solutions are proposed, such as the H-infinity control, that provides a transfer function more complex than the PI, and the LQR control, another classical control technique that imposes a state-feedback control action. Moreover, because the plant under control is characterized by a highly non-linear behavior and different model uncertainties, the SM control and the FL control are tested. The SM allows to apply a suitable control action without linearizing the plant, while the FL defines a "human-like" feedback that does not need of knowing the model of the plant either for design or for working properly. The design of all the proposed control techniques was achieved successfully, obtaining interesting results in the Simulink environment.

Keeping in mind the hardware implementation, the design of an EKF was taken into account. Because the measure of the inductor current is not available, the EKF appears to be the best suitable solution for estimating the state (fundamental for the LQR and the SM controllers) and for reducing the noise of the measurement stage. Also in this case, the design and the simulation of the filter were successful, providing a variation to the simple EKF with the cascade EKF, able to reduce the noise on the estimate of the inductor and ensuring smaller oscillations in the command activity of the state-feedback controllers.

Then, all the algorithms designed and tested in simulation were implemented in

hardware, for controlling a novel prototype of the NIBB converter. This necessity imposed different optimization problems, related to the limited capabilities of the hardware. As a matter of facts, the implementation of the PI and of the H-infinity controller was achieved without any particular issue: both the algorithms allow to control efficiently the prototype without giving time execution problems.

The implementation of the EKF and of the other control laws, instead, showed more difficulties in terms of execution and command activities. The EKF, due to the computation complexity provided by the algorithm, cannot work at the desired frequency of 30 kHz, reaching at most 4 kHz of execution. The filtering of the measures provide quite good results, however the estimate of the inductor current is not able to converge to the same value provided by the oscilloscope. For this reason, the SM and LQR are implemented using simple observers, based on the model of the plant, instead of exploiting the EKF. Due to the noise on the measures and the approximation applied for the inductor current, these algorithms are not able to work properly, providing a command input with high oscillations.

Moreover, due to the high number of conditions imposed by the 49 inference rules, the MCU was not capable of running the FL controller with a suitable working frequency.

In conclusion, this study provides a very deep insight on the different control possibilities that can be applied with DC-DC converters. The simulations show that all the tested control techniques may be successfully applied. However, the hardware represents an important limit that, for some cases, is difficult to outcome. The whole study shows as control based on transfer functions can be very effective, both in design and in implementation. As a matter of facts, the H-infinity design allows to take into account explicitly for different time requirements, giving as final result a transfer function able to provide the desired behavior and to ensure stability.

10.2 Future works

The improvements that can be carried out in future are mainly related to the hardware implementation:

- optimization of the EKF for achieving a working frequency of 30 kHz, that allows to have a fast convergence to the correct value of the state;
- optimization of the LQR and SM controllers, in order to achieve a command activity with reduced oscillations;
- optimization of the FL controller, trying to reduce the number of rules by means of any type of clustering.

Appendix A

Kalman filter theory

The EKF is the non-linear application of the Kalman filter theory, exploited in order to estimate the state and other possible unknown parameters/variables of a non-linear system. It is commonly called as observer and it can be seen as a virtual sensor, that is able to estimate variables which are not physically measured. This allows not only to save money and space (some sensors indeed are quite expensive, with an important bulk), but also to improve the accuracy of the measures provided by the physical sensors, through data fusion procedure.

The formulation of the EKF is provided directly in DT: this means that if the system is described by CT state equations, it has to be discretized with a suitable procedure (e.g. the Forward Euler discretization method). The starting point is the discrete state space representation given in equation (A.1):

$$\begin{aligned} x(k+1) &= f(x(k), u(k)) + v_1(k) \\ z(k) &= h(x(k)) + v_2(k) \end{aligned} \tag{A.1}$$

In this system, the state might be not known or partially unknown; v_1 and v_2 are unknown disturbances and u is the input of the model. The estimate provided by the Kalman filter is linked to some assumptions that, if not satisfied, may lead the observer to a very low accuracy:

1. v_1 and v_2 are uncorrelated white noises, such that:

$$\begin{aligned} E[v_1(k_1)v_1(k_2)^T] &= V_1\delta(k_2 - k_1) \\ E[v_2(k_1)v_2(k_2)^T] &= V_2\delta(k_2 - k_1) \\ E[v_1(k_1)v_2(k_2)^T] &= \mathbf{0}; \end{aligned}$$

2. f and h are known non-linear functions, V_1 and V_2 are known matrices;

3. the initial state $x(1)$ is an unknown random vector, with mean value equal to $\overline{x}_1$ and P_1 variance.

The formulation of the EKF does not change with respect to a linear Kalman filter: the main differences are related to the matrices that are used for computing the Kalman gain. Indeed, for a linear system, A and C matrices of the state space representation are exploited for the definition of the gain. However, these matrices are not defined for a non-linear system: the solution is to linearize the functions f and h in order to obtain suitable A and C matrices that can be employed in the algorithm.

$$\begin{aligned} \hat{A}(k|k) &= \left.\frac{\partial f}{\partial x}\right|_{\hat{x}(k|k)} \\ \hat{C}(k|k-1) &= \left.\frac{\partial h}{\partial x}\right|_{\hat{x}(k|k-1)} \end{aligned} \tag{A.2}$$

As shown in equation (A.2), $\hat{A}$ is obtained as linearization of f with respect to x around the filtered state, indicated as $\hat{x}(k|k)$. While $\hat{C}$ is the linearization of h with respect to x, around the predicted state, defined as $\hat{x}(k|k-1)$.

For the implementation of the EKF, the predictor/corrector form is employed, as indicated in A.3, providing a numerically reliable formulation of the equation involved in the observer.

$$\begin{aligned} &K_0(k) = P(k)\hat{C}(k)^T[\hat{C}(k)P(k)\hat{C}(k)^T + V_2]^{-1} && \textbf{Filter gain} \\ &P_0(k) = [I_n - K_0(k)\hat{C}(k)]P(k) && \textbf{Measurement update} \\ &e(k) = z(k) - h(\hat{x}(k), v(k)) && \textbf{Innovation} \\ &\hat{x}(k|k) = \hat{x}(k) + K_0(k)e(k) && \textbf{Corrector} \\ &P(k+1) = \hat{A}(k)P_0(k)\hat{A}(k)^T + V_1 && \textbf{Time update} \\ &\hat{x}(k|k-1) = f(\hat{x}(k|k), v(k), d(k)) && \textbf{Predictor} \end{aligned} \tag{A.3}$$

In equation (A.3), $\hat{A}(k) \equiv \hat{A}(k|k)$ and $\hat{C}(k) \equiv \hat{C}(k|k-1)$. They are written differently only for clarity sake.

Appendix B

Performance function

In order to compute the step performances related to the different simulations and hardware tests, a suitable Matlab transfer function is written, able to compute overshoot, rise time and settling time, with suitable graphical representations.

```
function [tr,ovs,ts] = time_performance(iin,iref,t,alpha,tol,plotcond,tau)
% [tr,ovs,ts] = time_performance(iin,iref,t,alpha,tol,plotcond,tau)
%
% Author: Arturo Rivelli
% Last update: 14/11/2021
%
% This function provides the computation of the rise time 'tr' (as tr90%),
% the percentage overshoot 'ovs' of the signal and the settling time 'ts'
% at 'alpha' percentage. It also provides representations of the signal
% with ts, tr and ovs specifications. This function can be used for every
% test of the NIBB DC-DC converter.
% Input arguments description:
%
% > iin: input current array
% > iref: reference current array. It is possible also to use an array with
%   just three elements: the starting step, the final step and the trigger
%   time [iref_0 iref_end t_ref]
% > t: time array
% > alpha: settling time percentage (set to 0 if not desired)
% > tol: rise time tolerance  (set to 0 if computed by the function)
% > plotcond: set to 'true' in order to print the plots
% > tau: starting time for plot (set to 0 if start time 0)
%

%% Tau condition
% If the 'tau' is 0 or the number of input arguments is less
% than 7, the whole time array is used
if nargin < 7 || tau == 0
    flagTau = false;
else
    flagTau = true;
end

%% Plot condition
% If the 'plotcond' is 'false' or the number of input arguments is less
% than 6, no plot is printed
if nargin < 6 || not(plotcond)
```

```
    flagPlot = false;
else
    flagPlot = true;
end

%% Tol condition
% If 'tol' is 0 or the number of input arguments is less
% than 5, the tolerance for the rise time is set as the distance between
% two points of the input current iin
if nargin < 5 || tol == 0
    tol_iin = abs(iin(2) - iin(1));
else
    tol_iin = tol;
end

%% Settling time condition
% If 'alpha' is 0 or the number of input arguments is less than 4, no
% settling time is computed
if nargin < 4 || alpha == 0
    ts = NaN;
    flagTs = false;
else
    flagTs = true;
end

%% Errors
% If there are less than 3 input variable, an error message is produced
if nargin < 3
    error("Not enough arguments")
end

% If iref is composed by less than 3 values or have different size than iin
% an error message is produced
if length(iref) ~= length(iin)
    if length(iref) ~= 3
        error("Invalid reference current array")
    end
end

%% Iref condition
% If iref is composed only by two elements, the whole reference current
% array is reconstructed
if length(iref) == 3
    iref_0 = iref(1); iref_end = iref(2);
    t_ref = iref(3);

    % iref build
    iref = zeros(length(t),1);
    iref_indx = find(t <= t_ref,1,'last');

    iref(1:iref_indx) = iref_0;
    iref(iref_indx+1:end) = iref_end;
end

%% Starting time
% If 'flagTau' is true, ts is computed
if flagTau
    % iin, iref and t are rearranged according to the start time tau
    tau_indx = find(t <= tau,1,'last');

    iin = iin(tau_indx:end);
    iref = iref(tau_indx:end);
    t = t(tau_indx:end);
```

```
end

% T0 is the step starting time
T0_indx = find(iref == iref(end),1);
T0 = t(T0_indx); t0 = t(T0_indx:end);

% Only the region after the rising edge is considered
iin_t0 = iin(T0_indx:end);

%% [tr] Rise time computation
% The step size is evalueted. Then the 90% of the step size is used as
% reference for tr90
stepSize = iref(end) - iref(1);
iref_90 = iref(1) + 0.9*stepSize;

% FindTr is a flag variable that is toggled to 1 when tr90 has been found
findTr = 0;
for i = 1:length(iin_t0)

    % Range of tolerance centerd in the actual current value iin(i)
    supVal = iin_t0(i) + tol_iin;
    infVal = iin_t0(i) - tol_iin;

    % Find tr90
    if iref_90 <= supVal && iref_90 >= infVal
        tr_90 = t0(i);
        findTr = 1;
        break
    end
end

if findTr
    % tr computation
    tr = tr_90 - T0;
else
    tr = NaN; tr_90 = NaN;
    warning("Rise time not found. tol = " + tol_iin + ...
        ". Increse the tolerance")
end

%% [ovs] Overshoot computation
% Max input current value
[iin_max,iin_max_indx] = max(iin_t0);

% Overshoot condition non negative
if iin_max > iref(end)

    % Overshoot flag
    flagOvs = 1;

    % Overshoot computation
    ovs = 100*(iin_max - iref(end))/iref(end);
    t_max = t(T0_indx + iin_max_indx);

else
    % Overshoot zero
    flagOvs = 0;
    ovs = 0;
end

%% [ts] Settling time computation
% If 'flagTs' is true, ts is computed
```

```
if flagTs
    % Conversion from percentage to decimal
    a = alpha/100;

    % Sup and inf limits definition
    iref_aSup = iref(end)*(1+a);
    iref_aInf = iref(end)*(1-a);

    % Flag that indicates if iin is inside the region defined by Sup and
    % Inf (i.e. the settling region)
    flagIn = false;

    for i = 1:length(iin_t0)

        % If iin(i) is outside settling region, 'flagIn' is false
        if iin_t0(i) < iref_aInf || iin_t0(i) > iref_aSup
            flagIn = false;
        end

        % If iin(i) is inside settling region, 'flagIn' is true and ts is
        % set. In this case, ts is set every time iin goes from 'outside'
        % to 'inside' the settling region.
        if not(flagIn)
            if iin_t0(i) >= iref_aInf && iin_t0(i) <= iref_aSup
                flagIn = true;
                ts_0 = t0(i);
                iin_set = iin_t0(i);
            end

        end

    end

    % WARNING: Check on the value of ts: if the alpha value chosen is too
    % small, ts is not computed.
    tol_t = 1e-5;
    t_inf = t(end)-tol_t;

    if ts_0 >= t_inf
        ts = NaN;
        warning("The value of alpha is " + alpha + ...
            ". Too small value. No valid ts can be computed")
    else
        % ts computation
        ts = ts_0 - T0;
    end

end

%% Plot printing
% The plot is printed only if flagPlot is true

if flagPlot

    % Plot colors
    blue = [0 0.4470 0.7410];
    orange = [0.8500 0.3250 0.0980];
    green = [0 1 0];

    % tr plot
    figure

    % iin vs iref
```

```
    plot(t,iin,'LineWidth',0.5,'Color',orange), hold on
    plot(t,iref,'LineWidth',1,'Color',blue), hold on

    % ovs
    if flagOvs
        yline(iin_max,'LineWidth',0.9,'Color',green), hold on
        xline(t_max,'--','LineWidth',0.9,'Color',green), hold on
        plot(t_max,iin_max,'^','MarkerSize',5,'LineWidth',1.1,'Color',green)
    end

    % iref_90 line
    yline(iref_90,'LineWidth',0.9,'Color',green), hold on

    if isfinite(tr_90)
        % tr_90 line
        xline(tr_90,'--','LineWidth',0.9,'Color',green), hold on

        % Rise point (tr_90,iref_90)
        plot(tr_90,iref_90,'o','MarkerSize',5,'LineWidth',1.1,'Color',green)
    end

    % Plot settings
    grid on, grid minor, xlim([t(1) t(end)])
    xlabel('\it Time [s]'), ylabel('\it Amplitude [A]')
    title("Rise time plot: t_r = " + tr*1000 + " ms, ovs = " + ...
        ovs + "%")

    % If 'flagTs' is true, also the ts plot is printed
    if flagTs

        % ts plot
        figure

        % iin vs iref
        plot(t,iin,'Color',orange), hold on
        plot(t,iref,'LineWidth',1.1,'Color',blue), hold on

        % iref_aInf and iref_aSup lines
        yline(iref_aInf,'Color',green), hold on
        yline(iref_aSup,'Color',green), hold on

        if isfinite(ts)
            % ts line
            xline(ts_0,'--','LineWidth',0.9,'Color',green), hold on

            % Settling point
            plot(ts_0,iin_set,'o','MarkerSize',5,'LineWidth',1.1,'Color',green)
        end

        % Plot settings
        grid on, grid minor, xlim([t(1) t(end)])
        xlabel('\it Time [s]'), ylabel('\it Amplitude [A]')
        title("Settling time plot: t_s = " + ts*1000 + ...
            " ms, \alpha = " + alpha + "%")

    end

end

end
```

Acronyms

TEG	Thermoelectric Generator
SMPS	Switching-Mode Power Supply
NIBB	Non Inverting Buck Boost
CCM	Continuous Current Mode
DCM	Discontinuous Current Mode
SISO	Single-Input Single-Output
MISO	Multi-Input Single-Output
ESR	Equivalent Series Resistor
MAE	Mean Absolute Error
RMSE	Root Mean Squared Error
RAE	Relative Absolute Error
RSE	Relative Squared Error
CT	Continuous Time
DT	Discrete Time
ACO	Ant Colony Optimization
PSO	Particle Swarm Optimization
ZOH	Zero-Order-Hold
LMI	Linear Matrix Inequality
LTI	Linear Time Invariant
LQR	Linear Quadratic Regulator
CARE	Continuous Algebraic Riccati Equation
DARE	Discrete Algebraic Riccati Equation
SM	Sliding Mode
FL	Fuzzy Logic
COG	Center of Gravity
EKF	Extended Kalman Filter
MCU	Micro-Controller Unit
FPU	Floating-Point Unit
TMU	Trigonometric Math Unit
SOC	Start of Conversion
GPIO	General Purpose Input/Output
IPC	Inter Processors Communication
CLA	Control Law Accelerator

List of Figures

List of Tables

Bibliography

[1] J. H. Carstens and C. Guhmann, "Adaptive control of a boost-buck converter for thermoelectric generators," in *2014 European Control Conference (ECC)*, (Strasbourg, France), pp. 2121–2126, IEEE, June 2014.

[2] C.-L. Wei, C.-H. Chen, K.-C. Wu, and I.-T. Ko, "Design of an Average-Current-Mode Noninverting Buck–Boost DC–DC Converter With Reduced Switching and Conduction Losses," *IEEE Transactions on Power Electronics*, vol. 27, pp. 4934–4943, Dec. 2012.

[3] A. Riccobono and E. Santi, "Comprehensive Review of Stability Criteria for DC Power Distribution Systems," *IEEE Transactions on Industry Applications*, vol. 50, pp. 3525–3535, Sept. 2014.

[4] R. W. Erickson and D. Maksimovic, *Fundamentals of Power Electronics*. Springer US, second ed., 2001.

[5] R. M. Kotecha, *Analysis and Comparison of Popular Models for Current-Mode Control of Switch Mode Power Supplies*. MSc Thesis, Wright State University, 2010.

[6] A. Benlafkih, S.-d. Krit, and M. C. Elidrissi, "A Comparative study of Analog and digital Controller On DC/DC Buck-Boost Converter Four Switch for Mobile Device Applications," *International Journal of Computer Science Issues*, vol. 10, June 2013.

[7] C.-W. Chang and C.-L. Wei, "Single-inductor four-switch non-inverting buck-boost dc-dc converter," in *Proceedings of 2011 International Symposium on VLSI Design, Automation and Test*, (Hsinchu, Taiwan), pp. 1–4, IEEE, Apr. 2011.

[8] N. Zhang, G. Zhang, and K. W. See, "Systematic Derivation of Dead-Zone Elimination Strategies for the Noninverting Synchronous Buck–Boost Converter," *IEEE Transactions on Power Electronics*, vol. 33, pp. 3497–3508, Apr. 2018.

[9] Young-Joo Lee, A. Khaligh, and A. Emadi, "A Compensation Technique for Smooth Transitions in a Noninverting Buck–Boost Converter," *IEEE Transactions on Power Electronics*, vol. 24, pp. 1002–1015, Apr. 2009.

[10] K. Muro, T. Nabeshima, T. Sato, K. Nishijima, and S. Yoshida, "H-bridge buck-boost converter with dual feedforward control," in *2009 International Conference on Power Electronics and Drive Systems (PEDS)*, (Taipei), pp. 1002–1007, IEEE, Nov. 2009.

[11] R. Keskin and I. Aliskan, "Design of Non-Inverting Buck-Boost Converter for Electronic Ballast Compatible with LED Drivers," *Karaelmas Science and Engineering Journal*, pp. 473–481, Dec. 2018.

[12] I. Aharon, A. Kuperman, and D. Shmilovitz, "Analysis of Dual-Carrier Modulator for Bidirectional Noninverting Buck–Boost Converter," *IEEE Transactions on Power Electronics*, vol. 30, pp. 840–848, Feb. 2015.

[13] Xiaoyong Ren, Xinbo Ruan, Hai Qian, Mingqiu Li, and Qianhong Chen, "Three-Mode Dual-Frequency Two-Edge Modulation Scheme for Four-Switch Buck–Boost Converter," *IEEE Transactions on Power Electronics*, vol. 24, pp. 499–509, Feb. 2009.

[14] M. F. N. Tajuddin, N. A. Rahim, I. Daut, B. Ismail, and M. F. Mohammed, "State space averaging technique of power converter with digital PID controller," in *TENCON 2009 - 2009 IEEE Region 10 Conference*, (Singapore), pp. 1–6, IEEE, Nov. 2009.

[15] V. M. S. Rodriguez, *PID Controller Tuning and Adaptation of a Buck Converter*. Doctoral Dissertation, Arizona State University, Aug. 2016.

[16] F. Ruiz, C. Fuentes, F. Flores-Bahamonde, J. Calvente, R. Giral, and C. Restrepo, "Digital current control of the versatile buck-boost converter for photovoltaic applications," in *2017 IEEE 8th International Symposium on Power Electronics for Distributed Generation Systems (PEDG)*, (Florianopolis, Brazil), pp. 1–6, IEEE, Apr. 2017.

[17] M. Karimi, A. Abrishamifar, and M. Fazeli, "DSP-Based integrated control modeling and implementation of Noninverting Buck-Boost Converter," *Bulletin de la Société Royale des Sciences de Liège*, vol. 85, p. 12, 2016.

[18] M. Jokinen, M. Niemela, and J. Pyrhonen, "Bump-less transfer algorithm between two different velocity controllers in an electric drive," in *International Symposium on Power Electronics, Electrical Drives, Automation and Motion, 2006. SPEEDAM 2006.*, (Taormina, Italy), pp. 685–690, IEEE, 2006.

[19] A. M. Bozorgi, V. Fereshtehpoor, M. Monfared, and N. Namjoo, "Controller Design Using Ant Colony Algorithm for a Non-inverting Buck–Boost Chopper Based on a Detailed Average Model," *Electric Power Components and Systems*, vol. 43, pp. 177–188, Jan. 2015.

[20] E. Sahin, "A PSO Tuned Fractional-Order PID Controlled Non-inverting Buck-Boost Converter for a Wave/UC Energy System," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 4, pp. 32–37, Dec. 2016.

[21] H. K. Khleaf, A. K. Nahar, and A. S. Jabbar, "Intelligent control of DC-DC converter based on PID-neural network," *International Journal of Power*

Electronics and Drive Systems (IJPEDS), vol. 10, p. 2254, Dec. 2019.

[22] R. Dowlatabadi, M. Monfared, S. Golestan, and A. Hassanzadeh, "Modelling and controller design for a non-inverting buck-boost chopper," in *Proceedings of the 2011 International Conference on Electrical Engineering and Informatics*, (Bandung, Indonesia), pp. 1–4, IEEE, July 2011.

[23] C.-W. Chen, K.-H. Chen, and Y.-M. Chen, "Modeling and controller design for a four-switch buck-boost converter in distributed maximum power point tracking PV system applications," in *2012 IEEE Energy Conversion Congress and Exposition (ECCE)*, (Raleigh, NC, USA), pp. 1663–1668, IEEE, Sept. 2012.

[24] C. Olalla, I. Queinnec, R. Leyva, and A. El Aroudi, "Optimal State-Feedback Control of Bilinear DC–DC Converters With Guaranteed Regions of Stability," *IEEE Transactions on Industrial Electronics*, vol. 59, pp. 3868–3880, Oct. 2012.

[25] H. Ramirez, G. Garzón, C. Torres, J. Navarrete, and C. Restrepo, "LMI Control Design of a Non-Inverting Buck-Boost Converter: A Current Regulation Approach," *TECCIENCIA*, vol. 12, pp. 79–85, Apr. 2017.

[26] K. Zhou and J. C. Doyle, *Essentials of Robust Control*, vol. 104. Prentice Hall, 1998.

[27] L. Mohammadian, E. Babaei, and M. B. Bannae Sharifian, "Buck-Boost DC-DC Converter Control by Using the Extracted Model from Signal Flow Graph Method," *International Journal of Applied Mathematics, Electronics and Computers*, vol. 3, p. 155, June 2015.

[28] F. Leung, P. Tam, and C. Li, "The control of switching DC-DC converters-a general LWR problem," *IEEE Transactions on Industrial Electronics*, vol. 38, no. 1, pp. 65–71, Feb./1991.

[29] C. Olalla, R. Leyva, A. El Aroudi, and I. Queinnec, "Robust LQR Control for PWM Converters: An LMI Approach," *IEEE Transactions on Industrial Electronics*, vol. 56, pp. 2548–2558, July 2009.

[30] P. Barbosa, L. Junior, R. Valle, A. Ferreira, and V. Montagner, "A Lqr Design With Rejection Of Disturbances And Robustness To Load Variations Applied To A Buck Converter," *Eletrônica de Potência*, vol. 21, pp. 7–15, Feb. 2016.

[31] H. Dehghani, A. Abedini, and M. Tavakoli Bina, "Advanced non-inverting step up/down converter with LQR control technique," in *4th Annual International Power Electronics, Drive Systems and Technologies Conference*, (Tehran, Iran), pp. 254–260, IEEE, Feb. 2013.

[32] B. D. O. Anderson and J. B. Moore, *Optimal Control: Linear Quadratic Methods*. Prentice-Hall Information and System Sciences Series, Englewood Cliffs, N.J: Prentice Hall, 1990.

[33] Z. Chen, J. Hu, and W. Gao, "Closed-loop analysis and control of a non-inverting buck–boost converter," *International Journal of Control*, vol. 83, pp. 2294–2307, Nov. 2010.

[34] M. Prist, E. Pallotta, P. Cicconi, P. Venturini, A. Monteriu, M. Germani,

and S. Longhi, "Energy Saving in Industrial Wireless Power Recharge System: Simulation of a PI-Sliding Mode Control for a Non-Inverting Buck-Boost Converter," in *2018 IEEE PELS Workshop on Emerging Technologies: Wireless Power Transfer (Wow)*, (Montréal, QC), pp. 1–6, IEEE, June 2018.

[35] M. Agostinelli, R. Priewasser, S. Marsili, and M. Huemer, "Fixed-frequency Pseudo Sliding Mode control for a Buck-Boost DC-DC converter in mobile applications: A comparison with a linear PID controller," in *2011 IEEE International Symposium of Circuits and Systems (ISCAS)*, (Rio de Janeiro, Brazil), pp. 1604–1607, IEEE, May 2011.

[36] S. K. Pandey, K. Veeranna, S. Patil, and S. Phadke, "Uncertainty Estimator based Sliding Mode Control Schemes for Multimode Noninverting Buck-Boost DC-DC Converter," *IFAC-PapersOnLine*, vol. 53, no. 1, pp. 555–560, 2020.

[37] M. A. Qamar, J. Feng, A. U. Rehman, and A. Raza, "Discrete time sliding mode control of DC-DC buck converter," in *2015 IEEE Conference on Systems, Process and Control (ICSPC)*, (Bandar Sunway, Malaysia), pp. 91–95, IEEE, Dec. 2015.

[38] J. Samantaray and S. Chakrabarty, "Digital Implementation of Sliding Mode Controllers with DC-DC Buck Converter System," in *2018 15th International Workshop on Variable Structure Systems (VSS)*, (Graz), pp. 255–260, IEEE, July 2018.

[39] K. Haninger and K. Hedrick, "Discrete-time implementations of sliding mode control," in *2016 American Control Conference (ACC)*, (Boston, MA, USA), pp. 6519–6524, IEEE, July 2016.

[40] C. Buccella, C. Cecati, H. Latafat, and K. Razi, "Digital control of a half-bridge LLC resonant converter," in *2012 15th International Power Electronics and Motion Control Conference (EPE/PEMC)*, (Novi Sad, Serbia), pp. LS6a.4–1–LS6a.4–6, IEEE, Sept. 2012.

[41] A. Hajizadeh, "Fuzzy/State-Feedback Control of a Non-Inverting Buck-Boost Converter for Fuel Cell Electric Vehicles," *Iranica Journal of Energy and Environment*, vol. 5, no. 1, 2014.

[42] Z. B. Duranay, H. Guldemir, and S. Tuncer, "Fuzzy Sliding Mode Control of DC-DC Boost Converter," *Engineering, Technology & Applied Science Research*, vol. 8, pp. 3054–3059, June 2018.

[43] O. N. Almasi, V. Fereshtehpoor, M. H. Khooban, and F. Blaabjerg, "Analysis, control and design of a non-inverting buck-boost converter: A bump-less two-level T–S fuzzy PI control," *ISA Transactions*, vol. 67, pp. 515–527, Mar. 2017.

[44] Z. Ortatepe and A. Karaarslan, "The Strategy of Battery Voltage Control using Hybrid PI+ANFIS Structure in Non-Inverting Buck-Boost Converter," in *International Symposium on Electrical Railway Transportation Systems, ERUSIS'2017*, Oct. 2017.

[45] S. J. Jawhar, N. Marimuthu, and N. A. Singh, "An Neuro-Fuzzy Controller for a Non Linear Power Electronic Boost Converter," in *2006 International*

Conference on Information and Automation, (Colombo, Sri Lanka), pp. 394–397, IEEE, Dec. 2006.

[46] R. M. Jaju and N. N. Kasat, "Neuro-Fuzzy Buck Boost Converter Implement on FPGA," *International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering*, vol. 2, p. 5, Jan. 2014.

[47] C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller. I," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, pp. 404–418, Mar. 1990.

[48] C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller. II," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, pp. 419–435, Mar. 1990.

[49] Han-Xiong Li and H. Gatland, "A new methodology for designing a fuzzy logic controller," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, pp. 505–512, Mar. 1995.

[50] A. Pereira, C. Duarte, P. Costa, and W. Gora, "Performance Improvement of a Buck Converter using Kalman Filtering," *International Journal of Power Electronics*, vol. 8, p. 87, 2017.

[51] M. Y. Candan and M. M. Ankarali, "Extended Kalman Filter Based State and Parameter Estimation Method for a Buck Converter Operating in a Wide Load Range," in *2020 IEEE Energy Conversion Congress and Exposition (ECCE)*, (Detroit, MI, USA), pp. 3238–3244, IEEE, Oct. 2020.

[52] Q. Tong, C. Chen, Q. Zhang, and X. Zou, "A Sensorless Predictive Current Controlled Boost Converter by Using an EKF with Load Variation Effect Elimination Function," *Sensors*, vol. 15, pp. 9986–10003, Apr. 2015.

[53] Q. Zhang, Q. Tong, and H. Zhang, "An Inductor Current Observer Based on Improved EKF for DC/DC Converter," in *2014 International Symposium on Computer, Consumer and Control*, (Taichung, Taiwan), pp. 892–895, IEEE, June 2014.