UNIVERSITA' DEGLI STUDI DI PADOVA

FACOLTA' DI INGEGNERIA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA ELETTRONICA

Tesi di laurea magistrale

Application of predictive control to a three-phase NPC converter: from simulation to a real platform

Laureando:

Nicola Bassan

Relatore:

Maria Elena Valcher

Correlatori:

Josep Bordonau, Alejandro Calle

ANNO ACCADEMICO 2011/2012

1. Summary

This final project treats an application of Predictive Control to a three levels inverter with a Neutral Point Clamped topology.

Load and converter models are used to predict current behaviour, and thereby select the most appropriate actuation following an arbitrary control criteria. Predictive control is a very wide concept and different control methods have been presented under this name.

The implementation of the control was carried out on an experimental platform used in previous projects present in the laboratories of the Grup de Recerca en Elèctronica de Potència in the UPC ETSEIB, Polytechnic University of Catalonia. This consists of two three-level converter with Neutral-Point Clamped topology in a back-to-back configuration, where only the inverter stage is used in this project, along with the equipment necessary for the control of the converter and for monitoring the desired electrical signals.

The main objective is to implement the control, first in a PC simulation and after in the real platform.

The project is divided into the following parts. Initially has been implemented a model of the system in a PC, performing simulations in different conditions. Aiming at the final realization of experimental tests, the model has been adapted to the platform. Finally, experimental tests have been performed, allowing to compare the theoretical behaviour with the real behaviour.

Table of Content

1. Glossary

C1, C2	→	*DC Bus Capacitors*
AC	→	*Alternative Current*
DC	→	*Direct current*
DEE	→	*Departament d'Electrònica de Potència.*
DSP	→	*Digital Signal Processor*
I/O	→	*Input/Output*
FPGA	→	*Field-Programable Gate Array*
GREP	→	*Grup de Recerca en Electrònica de Potència*
IGBT	→	*Insulated Gate Bipolar Transistor*
ISA	→	*Industry Standard Architecture*
NPC	→	*Neutral Point Clamped*
PPC	→	*PowerPC*
SRAM	→	*Static Random Access Memory*
FSC	→	*Finite Control Set*
MPC	→	*Model Predictive Control*

2. Preface

2.2. Project Origin

This project is part of the research conducted by the Research Group on Power Electronics (GREP) within the Department of Electronic Engineering (DEE) and focusing on the application of power electronic converters and multilevel converters, especially in renewable energies (mainly wind power and solar energy).

This project originates from the need to expand knowledge in a recent type of control that allows to use in a better way the possibilities offered by the development of powerful and fast microprocessors, looking specifically to improve the quality of electrical power obtained from high power generators.

3. Introduction

3.1. Project Objective

The main objective for this project is to apply a Predictive Control algorithm to a three-level NPC converter, in a way to achieve a required voltage at the output of this.

3.2. Project Scope

The following points define in more detail the scope of this project:

- Creation and simulation of a model for the application of the Predictive Control algorithm to a converter.
- Implementation of the model on a real platform.
- Compare experimental results with simulation results.

4. System Description

4.1. Full System

The equipment that has been used to obtain the experimental results of this project represent a part of a wind emulation platform, with a power converter of four quadrants. This equipment has been used to develop several final projects and to experimentally test the results of research carried out in the GREP.
In Fig. 4.1. one can see what are the elements that constitute the wind emulation platform. Items (3) and (4) are not used in this project (their function is to generate electricity with a synchronous generator (4), from the driving force created by the permanent magnet motor (4), controlled by the *back-to-back* commercial inverter Simovert (3)).

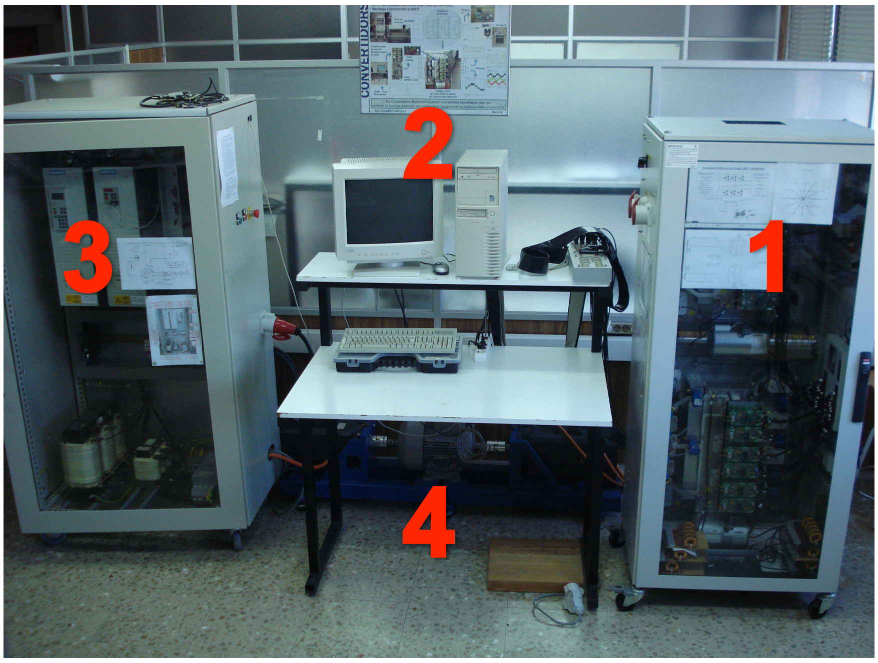
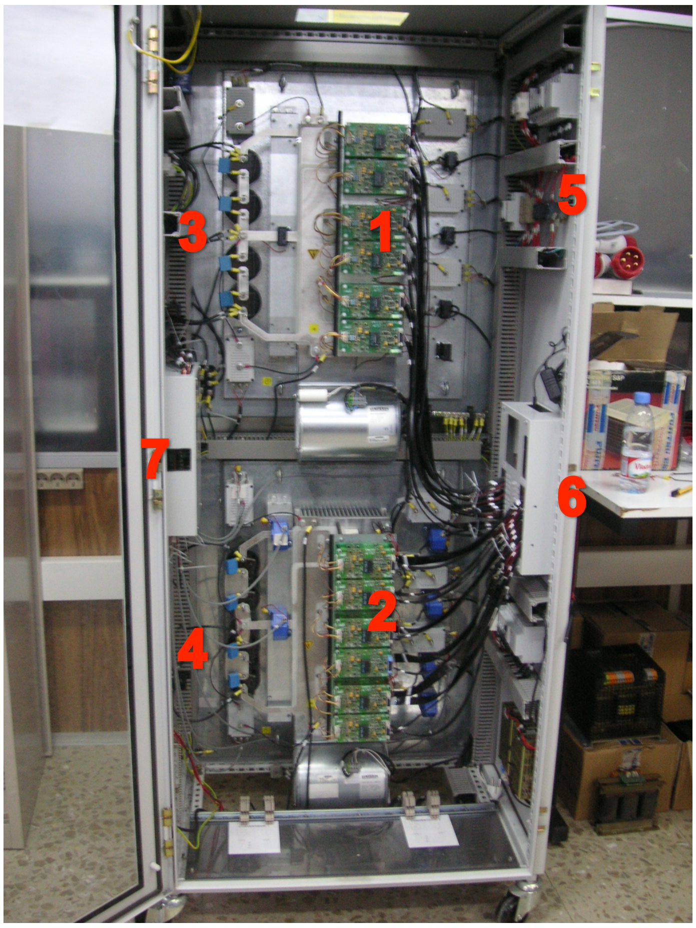


Fig. 4.1: Laboratory Equipment

Item (1) in Fig.4.1 is the converter cabinet and item (2) is the PC control center. These two elements are the equipment used for the realization of the experimental tests described in this project.

The cabinet of the converter (Fig. 4.2.), contains a four-quadrant *back-to-back* converter, built in previous final projects [1] and [2]. For this project, we have used only one of two converters that form the back-to-back configuration, specifically the one at the bottom. This was not chosen for any particular reason, either of them can operate as rectifier or inverter.



1- Top Inverter
2- Bottom Inverter
3- Top DC link capacitors
4- Bottom DC link capacitors
5- Control circuit
6- Digital Connections Board + FPGA
7- Analogic Sensors Board

Fig. 4.2: Converter Cabinet

In the current configuration, the converter becomes a two quadrants inverter able to deliver active and limited reactive power, thanks to capacitors of DC bus. Fig. 4.3 shows a diagram specifying the relationships between different subsystems used in the project.

The system control center is the PC. With this and the right software, it is possible to program the different subsystems involved in the control. In addition, there is a new user interface with which it is possible to modify in real time parameters and monitor signals of the inverter.

4.2. Converter Cabinet

In the cabinet are housed all the specific components for the power transmission and for the control of the inverter, in addition to protection circuits, start button and emergency stop, and connectors to the outside.

4.2.1. 3 Level Converter

Each of the two inverters is composed of 12 IGBT, Semikron SKM 100 GB/GAL/RAG. To control them 12 *drivers* are used, one for each IGBT, Semikron SKHI 10. The IGBTs are mounted on heat sinks, and user can select natural or forced by fans cooling.

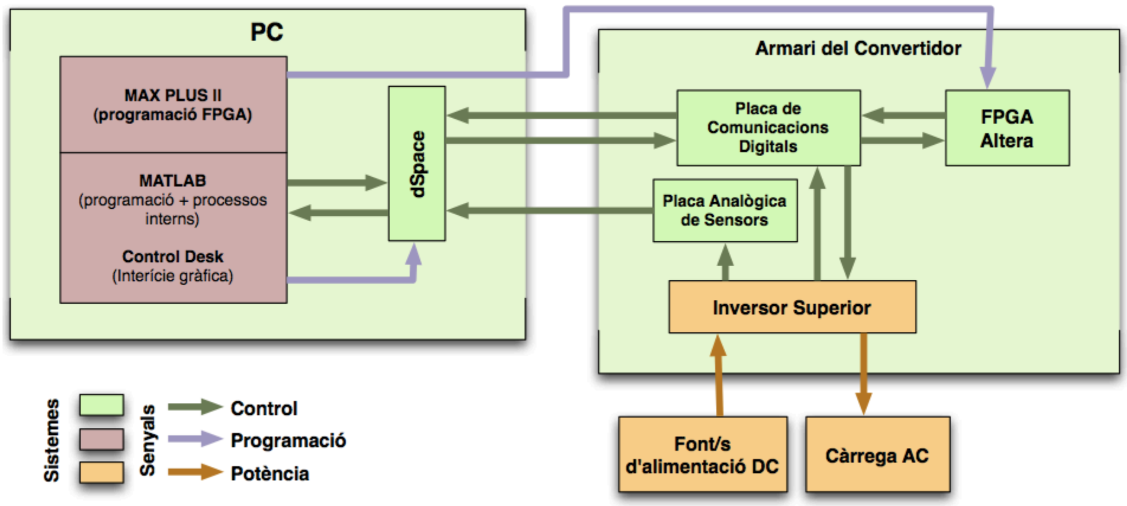

PC
MAX PLUS II (programació FPGA)
MATLAB (programació + processos interns)
Control Desk (Interície gràfica)
dSpace
Armari del Convertidor
Placa de Comunicacions Digitals
FPGA Altera
Placa Analògica de Sensors
Inversor Superior
Font/s d'alimentació DC
Càrrega AC
Sistemes
Senyals
Control
Programació
Potència

Fig. 4.3: Entire System Diagram

4.2.1.1. Topology - Three-phase DC/AC three level converters

In Figure 4.4 you can see the model used:

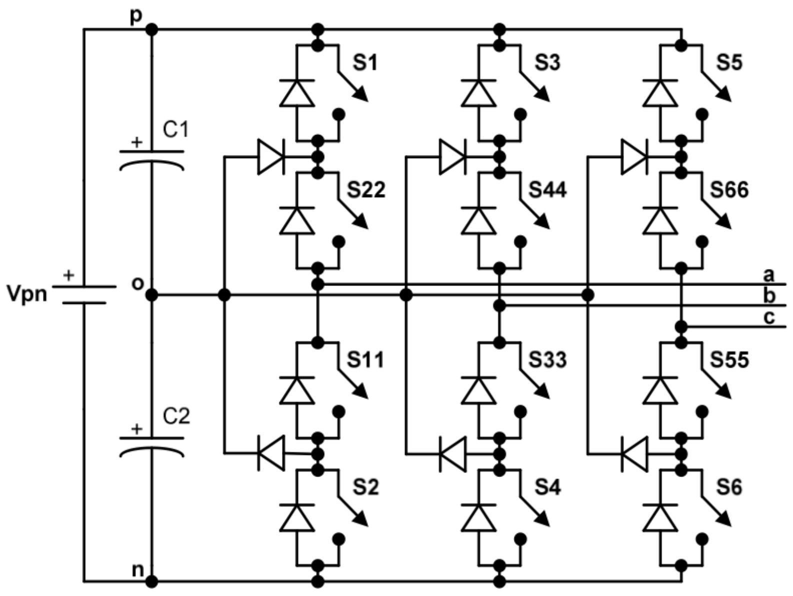


Fig. 4.4: Three phase DC/AC three level converter

The three-level converter topology used in this project is called Neutral-Point Clamped (NPC) and, although there are other topologies intended for very specific applications, it is the most important for the application of the predictive control.

As shown in Figure 4.4, the three-level converter doubles the number of switches (transistors) in every branch in respect to two-level converter. This brings the following advantages:

- Using the same transistors and keeping the power it is possible to double the working voltage of the DC source, meanwhile reducing the current through the converter, and extending the useful life of the components.

- By keeping constant DC voltage, the transistors blocking voltage is reduced by a half, thus making it possible to reduce its size.

  These advantages have positive effects on energy production: they help maintaining the existing power facilities, meanwhile reducing the costs because of the size reduction of components used.

For the description of the functionality of the converter, it is necessary to define the changing fuction S_{xy}:

$$S_{xy} = \begin{Bmatrix} 1, & \text{if phase } x \text{ is connected to } y \\ 0, & \text{on the other side} \end{Bmatrix} \tag{4.1}$$

with $\begin{matrix} x \in \{a,b,c\} \\ y \in \{p,o,n\} \end{matrix}$

Restrictions are needed, to avoid short-circuit conditions between p and n (eq. 4.2), or the condition of a floating phase (eq. 4.3), in which there is no a connection to one of the 3 levels of voltage.

$$S_{xp} + S_{xo} + S_{xn} \leq 1 \tag{4.2}$$

$$S_{xp} + S_{xo} + S_{xn} > 0 \tag{4.3}$$

The system consisting of the last two equations becomes:

$$S_{xp} + S_{xo} + S_{xn} = 1 \tag{4.4}$$

Looking on (4.1), possible combinations are 3^3. And by relating restrictions with possible configurations of the system show in Fig. 4.4 we obtain:

$$\begin{Bmatrix} S_{xp} = S_j \cdot S_{jj} \\ S_{xo} = S_{jj} \cdot S_{kk} \\ S_{xn} = S_k \cdot S_{kk} \end{Bmatrix} \tag{4.5}$$

with $\begin{matrix} x \in \{a,b,c\} \\ j \in \{1,3,5\} \\ k \in \{2,4,6\} \end{matrix}$

It is necessary note that equations are valid for equivalent phases, meaning:

$$\begin{matrix} x = a \rightarrow j = 1, k = 2 \\ x = b \rightarrow j = 3, k = 4 \\ x = c \rightarrow j = 5, k = 6 \end{matrix} \tag{4.6}$$

4.2.2. Altera UP2 Board (FPGA)

The UP2 board (Fig. 4.5) is a printed circuit board incorporates two integrated circuits that develop functions of a programmable logic device. The board is produced by Altera and is designed for academic use.

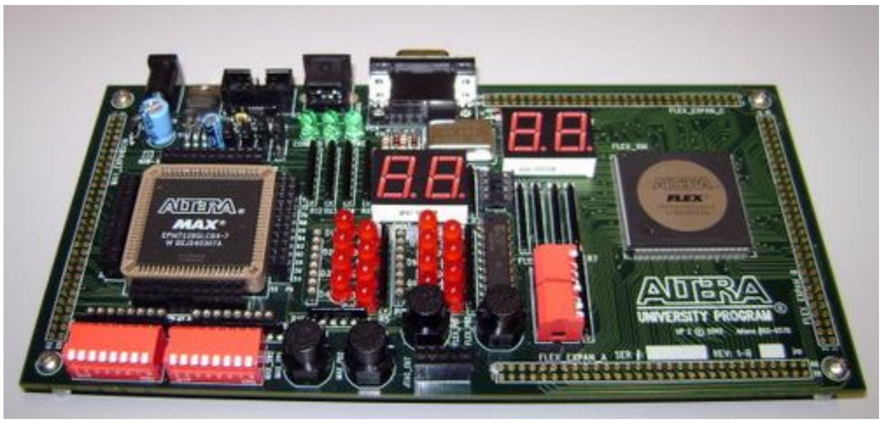


Fig. 4.5: Altera FPGA UP2 Board

Being a programmable logic device, it can be programmed by the user to perform any kind of desired function, restricted to a digital level, due to the fact that this board does not have analogic inputs or A/D converters.

This board is responsible for post-processing signals received from the DSpace, which control the activation of IGBTs by their corresponding drivers.

4.2.2.1. Hardware

The UP2 board incorporates two programmable devices, the EPM7128S of the family MAX (PLD), and the EPF10K70 of the family FLEX10K (FPGA), commanded by a 25,125 MHz clock signal.

Due to the fact that the functions carried out with the UP2 board do not require much computing power, only one of two integrated circuits of the FPGA is used, specifically EPF10K70.

The EPF10K70 is based on SRAM technology, 240-pin package and contains 70,000 logic gates, 3744 Logic Element, LE, and 9 Embedded Array Block, EAB.

Because of the SRAM FPGA technology, this must be reprogrammed every time the board power is interrupted.

The programming of the devices is done via a JTAG connector that plugs into the parallel port of the PC.

4.2.2.2. Software

The MAX PLUS II Baseline v10.2 software was used for the design of combinational and sequential circuits (prepared in a modular and hierarchical structures) that define the operation functioning of the FPGA. To program the device it is possible to use different schematic or hardware description languages, such as the VHDL.

4.2.3. Digital Connections Board

The digital connections board, designed and modified in previous projects, allows communications between DSpace, FPGA and IGBT drivers.

The details of these connections are:

• **DSpace → FPGA**: signals that indicate at which level of voltage each branch needs to be connected at that time are sent. Also control signals related to the on-off command of the converter and the value of the blanking time are transmitted.

• **FPGA → DSpace**: signal error related to malfunctioning of IGBT or drivers, or an incorrect previously data shipment.

• **FPGA → IGBT Drivers**: Switching Signals for all IGBT.

• **IGBT Drivers → FPGA**: Signal error related to malfunctioning of the driver or IGBTs.

This board is connected to two +15 V power supplies NPL65 from Artesyn Technologies, that power on drivers. Also a +5V voltage is extracted using a voltage regulator circuit, for supply of TTL buffers that adapt signals sent to drivers.

4.3. Control Center (PC)

A generic PC running on Pentium II 300 MHz and Windows 98 is used. The system has not been updated (it has over 10 years) because of dSpace interface is an ISA bus (Industry Standard Architecture), and motherboards with this type of bus are no longer manufactured.

4.3.1. dSpace

dSpace board (Fig. 5.8.) is a printed circuit controlled by a PowerPC (PPC), fits inside the PC (using an ISA bus) that performs the calculation functions of the system.

4.3.1.1. Hardware

dSpace board used (DS1103) contains two DSP (Digital Signal Processor) with a master/slave configuration. For the master function is used a 604e PowerPC, and for slave function a Texas Instruments TMS320F240.

Fig. 4.6 shows a block diagram of the internal structure and peripherals of dSpace 1103. As it can be seen, it has a lot of input/output and internal functions, but for this project only Digital I/O of *Master PPC*, and Interruptions I/O will be used.

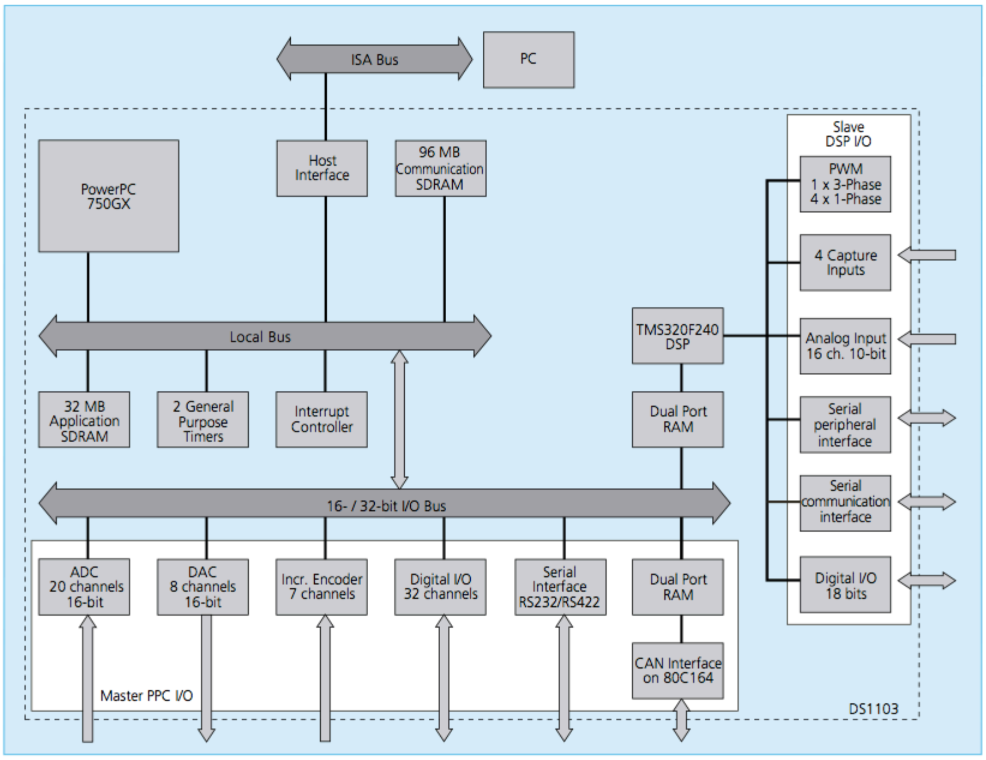


Fig. 4.6: Block Diagram that describes Internal Structure of dSpace Board 1103

4.3.1.2. Software

We can divide the software involved in the dSpace programming and real-time operations into three groups, each of them with a specific role.

- **MATLAB-Simulink**: this high-level programming software allows create programs using intuitive block diagrams (Simulink) and a simplified code. In addition, it is possible to simulate the performance of a model based on the inverter, thus allowing study the behaviour of this and adjust the control parameters, before the experimental tests. There are also a lot of built-in functions (*Toolboxes* and *Blocksets*) that simplify programming.

- ***Real-Time Workshop***: built-in MATLAB compiler, which convert the program implemented in Simulink to C code, compatible with DSpace. To run it, it is used the *Build* function, accessible from the menu bar of Simulink.

- ***Control Desk***: software that enables the design of interfaces for the user, using text boxes, graphics, indicators, etc. It is used for real-time monitoring of several variables of the control program running on the dSpace. It also allows the user to interactively modify different control variables, without re-programming the control card.

5. Predictive Control Description

5.1. Introduction

Current control of a three-phase converter is one of the most important and classical subjects in power electronics and it has been extensively studied in the last decades. Nonlinear methods, like hysteresis control and linear methods, like proportional-integral controllers using pulse width modulation (PWM) are well documented in literature [6].

Predictive control is a control theory that was developed at the end of the 1970s. Variants of this type of control strategy have found application in power converters. Predictive control has been used in current control, drives, power factor correction, and active filters. All of these applications consider linear models and use modulation techniques for voltage generation. As classic solutions, the basic idea under these methods is to consider the converter as a linear system instead of taking advantage of the discrete nature of the inverter and its control processor.

Behind the simple expression “predictive control” there is a very wide variety of different control methods, and a lot of applications have been presented under this name. Since the inverter has only a finite number of switching elements, the number of possible switching states is limited as well. For each of these switching states an equivalent circuit of the drive system without switching elements can be defined. Therefore the behaviour of inverter can be calculated separately and in advance for each of the switching states. Comparing the results of the calculation with the desired behaviour of the system, the optimal switching state of the inverter can be derived. A comparison between the precalculated and the real values at the end of the switching cycle allows the correction of model errors. Than the next switching state will be calculated using the corrected values.

One advantage of predictive control is the possibility to include nonlinearities of the system in the predictive model, and hence calculate the behaviour of the variables for different conduction states. This property was exploited in an earlier study reference, where predictive control was used to minimize switching frequency for high-power inverters. Also, this property of predictive control is used to evaluate the behaviour of the current error for each switching state in a single-phase active filter [4].

A conceptually different approach is presented to control a matrix converter. The model of the system is used to predict the behaviour of the load and input current for each different switching state of the matrix converter. The switching state that minimizes a cost function is selected. This method demonstrates that the use of predictive control can avoid the use of complex modulation techniques.

This project presents this method applied to a three-phase NPC inverter.

5.2. Description of Predictive Current Control

5.2.1. Operating Principle

The power converter or drive control problem can be defined as the determination of an appropriate control action $S(t)$ (usually the gate signals of the converter) that will drive a generic system variable $x(t)$ as close as possible to a desired reference value $x^*(t)$. Consider the qualitative behaviour of $x(t)$ and its regularly sampled value $x(t_k)$ with sampling period T_S for a system with a finite number of control actions n, as shown in Fig.5.1, where measurements, computations, and control actions are performed instantly (ideal case) [7].

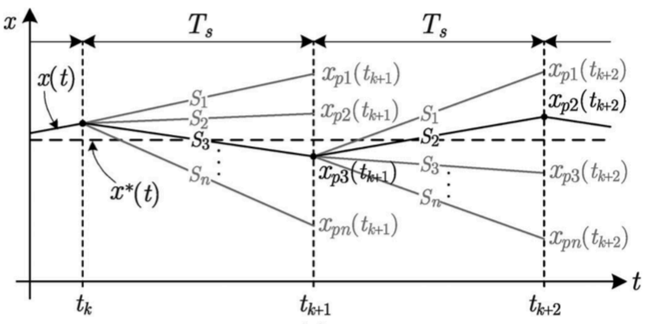


Fig. 5.1: FCS-MPC operating principle. Ideal theoretical case. [7]

Since the possible control actions are finite: S_i, with $i=1,...,n$, they can be evaluated together with the measured value $x(t_k)$, based on a prediction function f_P, to predict all the possible system transitions $x_{pi}(t_{k+1})=f_P\{x(t_k),S_i\}$, for $i=1,...,n$. This prediction function is directly derived from the discrete model and parameters of the system. To determine which of the control actions has to be selected, a decision or cost function f_g can be defined, usually dependent on the desired reference value and the predictions $g_i=f_g\{x^*(t_{k+1}),\ x_{pi}(t_{k+1})\}$, for $i=1,...,n$. Note that the future reference value is needed $x^*(t_{k+1})$, which can be assumed to be equal to the actual value $x^*(t_k)$, since T_S is sufficiently small compared with the dynamic behaviour of the system, and thus, the reference can be considered constant over T_S. If needed, the future reference value $x^*(t_{k+1})$, can be estimated via appropriate extrapolation methods.

A typical example for f_g would be the absolute error between the predictions and the reference $g_i = \left| x^*(t_{k+1}) - x_{pi}(t_{k+1}) \right|$. The evaluation of the cost function with corresponding to the n predictions will lead to n different costs. Naturally, the control action leading to the minimum cost ($\min\{g_i\}$, for $i=1,...,n$) is selected to control the system.

Based on the example shown in Fig.5.1, the predicted value $x_{p3}(t_{k+1})$ is the closest to the reference $x^*(t_{k+1})$; hence, S_3 is selected and applied at $t = t_k$. Following the same criterion, S2 is selected and applied at $t = t_{k+1}$. However, the ideal theoretical case in which the variables can be measured, predicted, and controlled instantaneously in $t = t_k$ is not possible in real-time applications. Nevertheless, this problem can be overcome if a two-step-ahead prediction is considered, as shown in Fig.5.2, in which the control action to be applied to the following sample time $S(t_{k+1})$ is determined. This way, a complete sampling period T_S is available to perform the algorithm. Naturally, the sampling period T_S has to be greater than the sum of the measurement, computation, and actuation times.

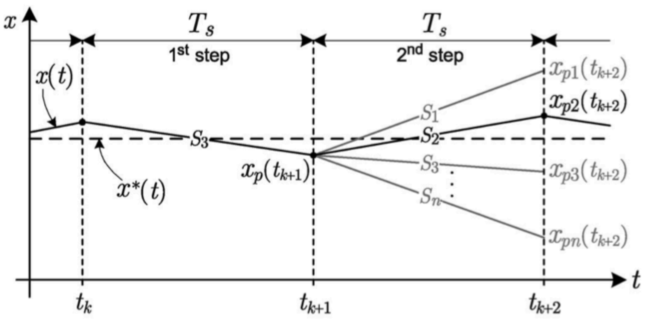


Fig. 5.2: FCS-MPC operating principle. Possible implementation case. [7]

It is worth mentioning that this control method is not limited to a single variable; on the contrary, multiple variables, system constraints, perturbations, saturations, and, basically, every characteristic that can be mathematically modelled and measured can be included in the predictive model and cost function. This is the basis of the great flexibility and control potential that can be achieved with FCS-MPC (Finite Control Set Model Predictive Control). Moreover, the fact that power converters have a reduced and limited number of switching states (or control set-ups) makes this method feasible to be implemented by means of present-day available microprocessing resources. Since only a discrete model of the system is necessary, rather than approximated linear models together with control system design theory and modulation algorithms, simpler and more direct design and implementation of the controller can be achieved.

In this project only the one-step prediction will be implemented, and the two-step-ahead prediction will be left for future development.

5.2.2. Control Strategy

The proposed predictive control strategy is based on the fact that a static power converter can generate only a finite number of possible switching states and that models of the system can be used to predict the behaviour of the variables for each switching state. For the selection of the appropriate switching state to be applied, a selection criterion must be defined. This selection criterion is expressed as a cost function that will be evaluated for the predicted values of the variables to be controlled. Prediction of the future value of these variables is calculated for each possible switching state. The switching state that minimizes the cost function is selected.

This control strategy can be summarized in the following steps [4]:

- Define a quality function *g*;
- Build a model of the converter and its possible switching states;
- Build a model of the load for prediction.

A discrete-time model of the load is needed to predict the behaviour of the variables evaluated by the quality function, i.e., the load currents.

A block diagram of the predictive control strategy applied to the current control for a three-phase inverter is shown in next figure.

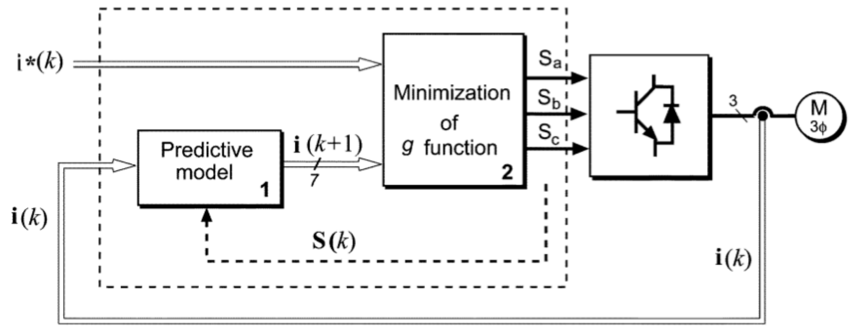


Fig. 5.3: Predictive current control method.

The current control is performed according to the following steps.

1) The value of the reference current is obtained (from an outer control loop), and the load current is measured.

2) The model of the system (block 1) is used to predict the value of the load current in the next sampling interval for each of the different voltage vectors.

3) The cost function evaluates the error between reference and predicted currents in the next sampling interval. The voltage that minimizes the current error is selected and applied to the load (block 2).

5.2.3. Model of the System

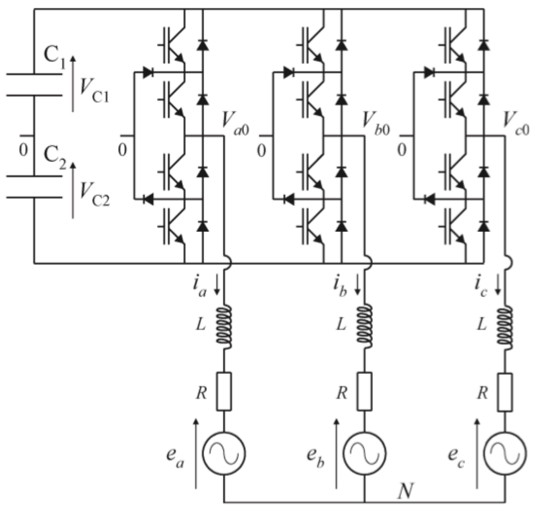


Fig. 5.4: Circuit of a three-phase NPC inverter connected to a resistive–inductive-active load. [5]

Fig. 5.1 shows a model of the system. It includes a three-phase three-level inverter and a resistive–inductive–active load. The reason for using this load is because it represents one of the most common applications for this kind of converter, an induction machine. Also, with this model, it is possible to evaluate a wide range of applications, including passive loads and grid-connected converters. The source of the reference current will depend on the specific application. For example, for field-oriented control of an induction machine, the reference current is generated by speed and flux controllers.

The converter applies to the load 19 voltage vectors, which are generated from 27 switching states, as presented in Fig. 5.2.

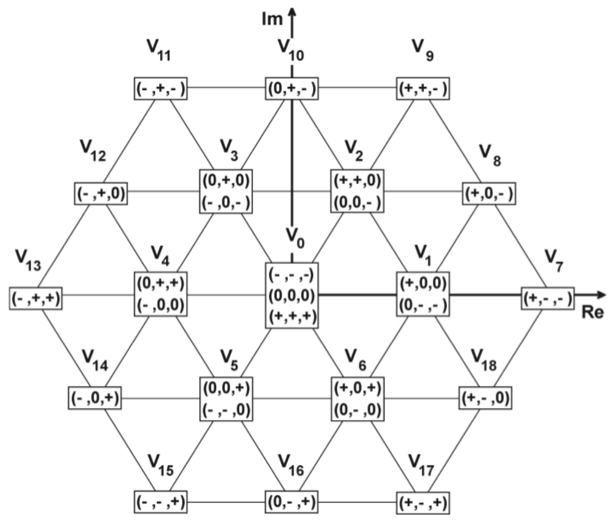


Fig. 5.5: Possible voltage vectors and switching states generated by a three-level invertir [5]

The center of an MPC algorithm is the model of the plant for which predictions are obtained. In this case, it corresponds to the equation of a three-phase resistive–inductive–active load, which fulfils [5]:

$$L\frac{d\mathbf{i}(t)}{dt} = \mathbf{v}(t) - R\mathbf{i}(t) - \mathbf{e}(t) \tag{5.1}$$

where *R* and *L* are the load resistance and inductance, respectively, **v** is the voltage vector generated by the inverter, **e** is the electromotive force (EMF) of the load, and **i** is the load current vector. These vectors are defined as:

$$\begin{aligned} \mathbf{v} &= \frac{2}{3}(V_{a0} + aV_{b0} + a^2V_{c0}) \\ \mathbf{i} &= \frac{2}{3}(i_a + ai_b + a^2 i_c) \\ \mathbf{e} &= \frac{2}{3}(e_a + ae_b + a^2 e_c) \end{aligned} \tag{5.2,3,4}$$

where $a = e^{j(2\pi/3)}$.

Upon assuming as sampling period T_s, the derivative form *d*i*(t)/dt* can be approximated by

$$\frac{d\mathbf{i}(t)}{dt} \approx \frac{\mathbf{i}(k) - \mathbf{i}(k-1)}{T_S} \tag{5.5}$$

Replacing (5.5) in (5.2) and shifting the discrete time one step forward, the relation between the discrete-time variables can be described as.

$$\mathbf{i}(k+1) = \frac{T_S}{RT_S + L}\left[\frac{L}{T_S}\mathbf{i}(k) + \mathbf{v}(k+1) - \mathbf{e}(k+1)\right] \tag{5.6}$$

Equation (5.6) is used to obtain predictions for the future value of the load current **i***(k+1)*, considering all possible voltage vectors v generated by the inverter and measured current at the *k*th sampling interval.

The control strategy also uses an estimation of the future reference current. Depending on the sampling time applied and the computational constrains, the estimation can be obtained by a second-order extrapolation given by

$$\mathbf{i}^*(k+1) = 3\mathbf{i}^*(k) - 3\mathbf{i}^*(k-1) + \mathbf{i}^*(k-2) \tag{5.7}$$

or, for a sufficiently small sampling time and also to save computational efforts, it is possible to consider $\mathbf{i}^*(k+1) \approx \mathbf{i}^*(k)$; thus, no extrapolation is necessary.

The current prediction in (5.6) also requires an estimation of the future load back electro magnetic force (EMF) **e**(*k*+1). That value, which is analogue to the future reference current case, can be estimated using a second-order extrapolation from present and past values or assuming **e**(*k*+1)≈ **e**(*k*). As mentioned, that will depend basically on the sampling time and the platform used for implementation. In this project **e**(*k*+1)≈ **e**(*k*) is used.

Finally, each capacitor from the DC link satisfies the following dynamic equation:

$$V_c(k+1) = V_c(k) + \frac{1}{C} i_c(k) T_S \tag{5.9}$$

where $i_c(k)$ is the current through the capacitor, $v_c(k)$ is its voltage, and *C* is the capacitance. Currents through the capacitors are obtained based on the load currents and the present switching state; thus, no additional measurements are needed. Using (5.9), it is possible to obtain predictions for the future value of the capacitor's voltage based on its present current and voltage.

5.2.4. Quality Function

The current error for the next sampling instant can be expressed in orthogonal coordinates as follows:

$$g = \left| i_\alpha^* - i_\alpha^p \right| + \left| i_\beta^* - i_\beta^p \right|$$

where i_α^p and i_β^p are the real and imaginary part of the predicted load current vector **i**(k+1), while i_α^* and i_β^* are the real and imaginary part of the reference current vector **i***.

Different control criteria will be expressed by means of different quality functions. In this project, the absolute error is used for computational simplicity.

The future value of the load current and voltages in the capacitors are predicted for the 27 switching states generated by the inverter, by means of (5.6) and (5.9). For this purpose it is necessary to measure the present load current and voltages in the capacitors. After obtaining the predictions, the quality function *g* is evaluated for each switching state. The switching state that minimizes *g* is selected and applied during the next sampling period.

The used quality function has the following composition:

$$g = f(\mathbf{i}^*(k+1), \mathbf{i}(k+1)) + h(V_{c12}(k+1)) \tag{5.10}$$

The first term in (5.10) is dedicated to achieve reference tracking, quantifying the difference between the reference current and current prediction at the next sampling time, for a given switching state. In this project the used *f* is:

$$f(\mathbf{i}^*(k+1), \mathbf{i}(k+1)) = \left| i_\alpha^*(k+1) - i_\alpha(k+1) \right| + \left| i_\beta^*(k+1) - i_\beta(k+1) \right| \tag{5.11}$$

where i_α and i_β are the real and imaginary components of current vector **i**, respectively.

The objective of the second term in (5.10) is to take advantage of the state redundancy of a three-level inverter, from the fact that the tracking cost *f* depends only on the voltage vector selected. Its expression is:

$$h(V_{C12}(k+1)) = \lambda_{dc} \cdot \left| V_{C1}(k+1) - V_{C2}(k+1) \right| \tag{5.12}$$

This is a term proportional to the absolute difference between both capacitors' voltage predictions. So, a switching state that generates smaller differences will be preferred. λ_{dc} is a weighing factor.

5.2.5. Control Algorithm

Here the predictive control algorithm implemented is explained by means of a flow diagram.

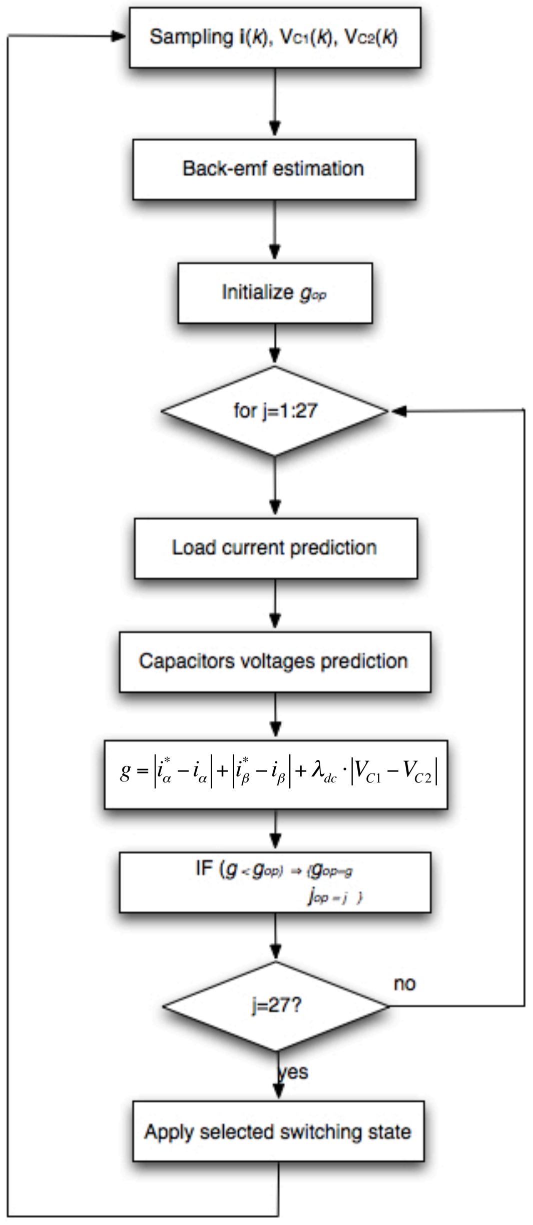

Sampling $\mathbf{i}(k)$, $V_{C1}(k)$, $V_{C2}(k)$
Back-emf estimation
Initialize g_{op}
for j=1:27
Load current prediction
Capacitors voltages prediction
$g = |i^*_\alpha - i_\alpha| + |i^*_\beta - i_\beta| + \lambda_{dc} \cdot |V_{C1} - V_{C2}|$
IF $(g < g_{op}) \Rightarrow \{g_{op} = g$
$j_{op} = j \}$
j=27?
no
yes
Apply selected switching state

Fig. 5.6: Flow diagram of the implemented control algorithm

For the explanation of how this algorithm has been implemented in Simulink and run in the dSpace please refer to Appendix A of this project. All the diagrams with the explanation of the function of every single block is present, helping to understand the complete functioning of the system.

6. Control Programming

For the implementation of the entire system it is necessary to develop the control at 3 different levels:

1. **PC:** performs the necessary software runs for the interaction between user and machine, allow adjusting all the parameters of the control. Thanks to the Control Desk it is possible to monitor all the signals and the changes in the real time parameters of the system.

2. **dSPACE:** where the real control runs, in real time and in a deterministic way, due to the independent microprocessor (PowerPC) and thanks to the software created using Simulink, MATLAB Code and C language models.

3. **FPGA:** is responsible of sending control signals to the drivers of the inverter, depending on information received from the dSpace. For the development VHDL language is used.

6.1. dSpace Programming

The presence of the independent microprocessor in the installed board of the dSpace, allows to reduce drastically the amount of work performed by the PC, and leaves to PowerPC the entire execution and handling of communication with FPGA and consequently with the drivers of the converter.

The programming of the system has been realized with Simulink v5.0, inside MATLAB v.6.5. The use of an old version of the software is due to the fact that the hardware available in the laboratory, i.e. the dSpace, need a ISA bus, available only in older PC, and this means the impossibility of use of a recent version of Matlab, since that a more powerful computer would be required.

In Fig.6.1 it is possible to see the upper structure of Simulink entire system.

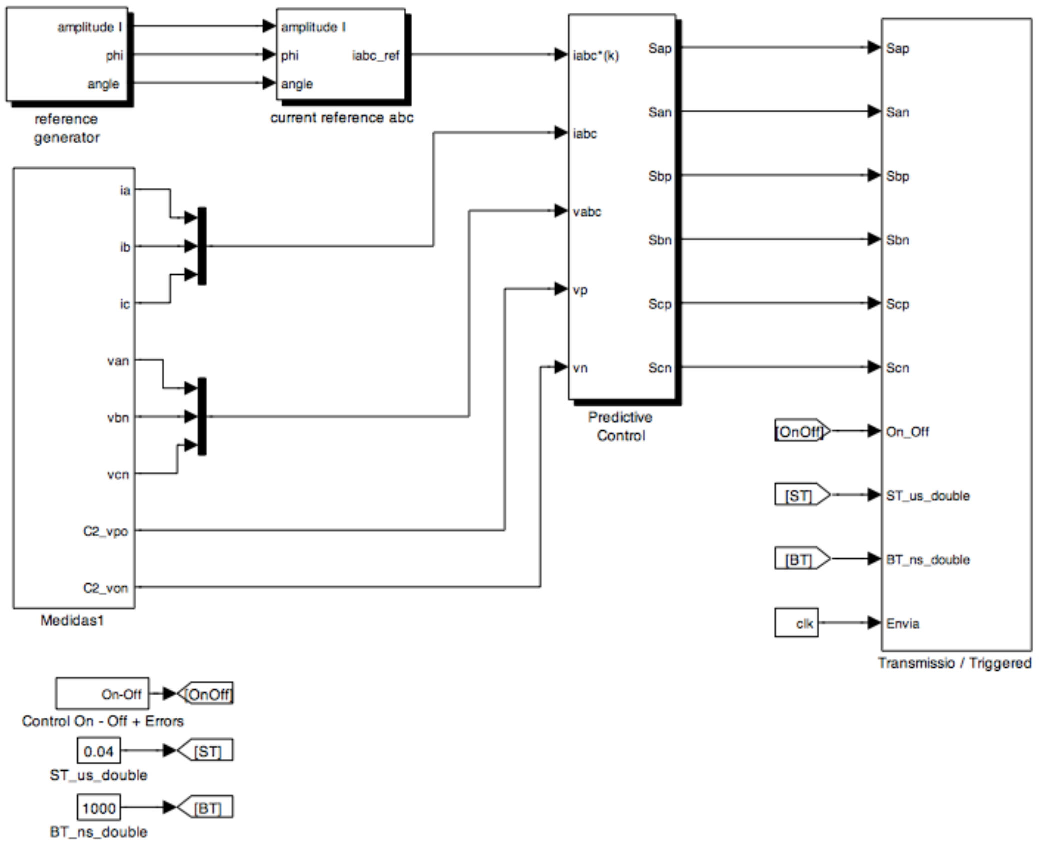


Fig. 6.1: Entire System Scheme

Main blocks of the model are now described.

6.1.1. Predictive Control Subsystem

It is the heart of the system, generates the configuration for the drivers. The inputs are the reference currents, the values measured on the 3-phases of current, voltage and the DC link capacitor voltages. The entire algorithm is implemented inside this block.

6.1.2. Trasmissio/Triggered Subsystem

The function C transmissió.c (DLL compiled file) selects sending data to be sent depending on on/off signal of the converter. In the first case (powered converter) the coded switching states of the converter are sent, and in the second (turned off converter) are sent the values of the *Clock Period* and of the *Blanking Time*. The value of the *on_off* signal (0 or 1, off or on respectively) is always sent.

The data are written to the 32-bit parallel bus used for digital I/O. These 32 bits are grouped into 4 bytes using predefined functions of the dSpace Blockset.

The transmission is made according to time schedule figure 6.2 and 6.3, depending on the state of the *on_off* signal:

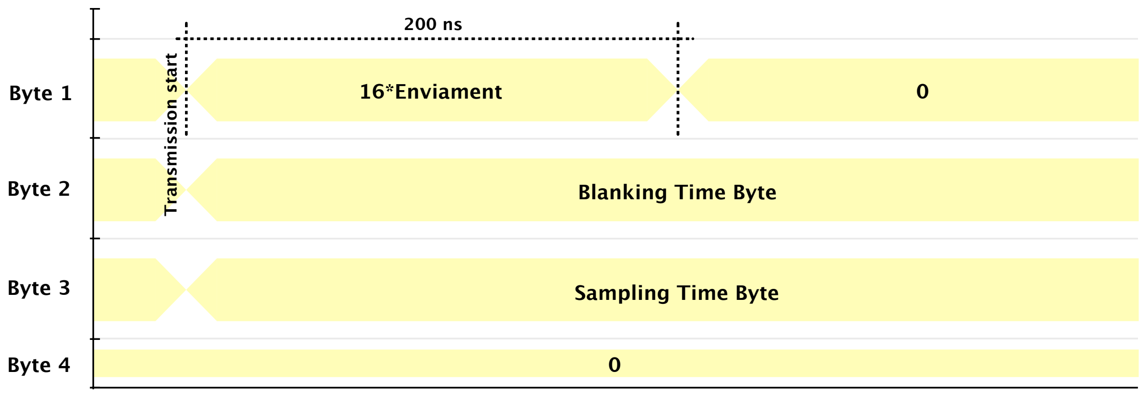


Fig. 6.2: Time Schedule of transmission - Converter Off

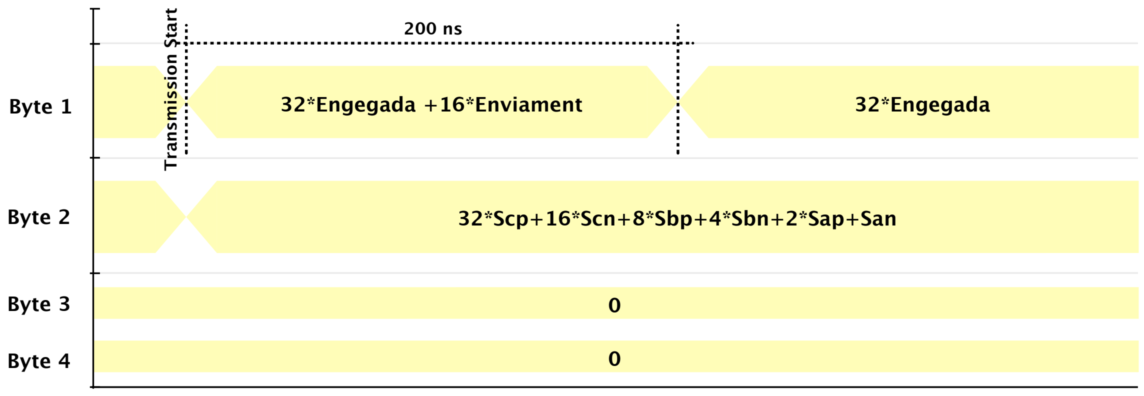


Fig. 6.3: Time Schedule of transmission - Converter On

6.2. dSpace – FPGA communication

An important point in the realization of the control was the choice of how to send data to the FPGA, since the different capacities of calculation and execution of the PowerPC and FPGA could create problems in terms of synchronization of the system.

3 different possibilities of sending the configurations to the drivers were analysed:

- Send a configuration every time that a change in the output was recognized.
- Create an hardware clock signal in the FPGA, and send the configuration present in the output every clock positive rising.
- Create a software clock, and send the configuration present in the output every clock positive rising.

Every possibility has been tested. The first had problems of configuration losses, meaning that sometimes a configuration was not sent to the drivers due to difference between software execution and effective transmission to the converter.

The second option worked, but the problem was the precision of the clock period, due to the delay introduced by the hardware execution in the FPGA.

So, third option was the best one, and the block named "*clk*" in the Simulink model creates a digital square signal with a period of 100μs, so that data are sent always with a frequency of 10 kHz.

The signal was not create using a Digital Clk Source Block of Simulink, but like shown in next figure:

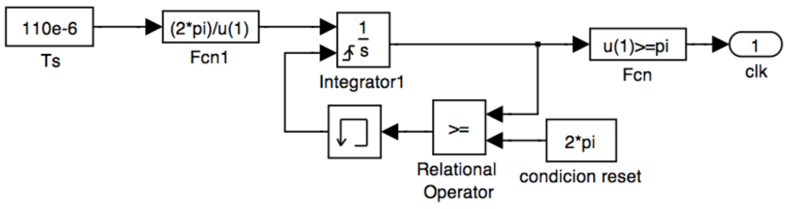


Fig. 6.4: *clk* Simulink block that creates a digital square signal

6.2.1. Adopted Solution

Next Figure shows all signals between dSpace and FPGA, with indication of direction of data.

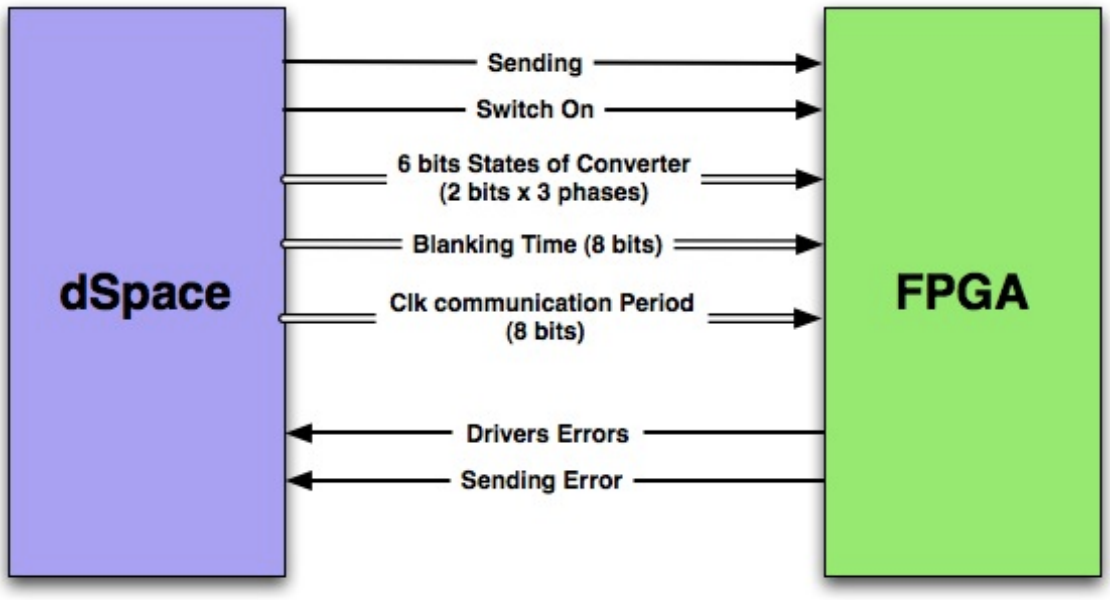

dSpace
FPGA
Sending
Switch On
6 bits States of Converter (2 bits x 3 phases)
Blanking Time (8 bits)
Clk communication Period (8 bits)
Drivers Errors
Sending Error

Fig. 6.5: Used signals for the communication between the dSpace and the FPGA

Now all the signals are described.

• **Sending**: Rising edge communicates to FPGA that new data is available in the data bus.

• **Switch On**: Logic signal that commands the switching on of the converter.

• **States of Converter**: The converter is formed by 12 IGBT, 4 in every branch, but the total number of configurations available in every phase can be codified with only 2 bits. So, no more than 6 bits are necessary for the entire system.

• **Blanking Time**: this is the time that the FPGA must wait before changing the state of IGBT following the order given by the dSpace.

• **Clk Communication Period**: This is the clock signal that commands the FPGA for the change of states.

• **Drivers Error**: Signal that indicates the activation of an alarm in one of the 6 drivers of IGBT (short circuit of IGBT or no supply of drivers).

• **Transmission Error**: activated when errors in transmission are detected.

7. Simulation and Experimental Results

For the correct evaluation of test results it was decided to compare the results obtained in the laboratory with those carried out through computer simulations. It was therefore necessary to establish criteria that can properly validate the different tests.

The system was tested in 2 different situations, the first using a resistive load, and, due to the positive result obtained, secondly by connecting the output of the inverter directly to the grid. To do this, it was necessary to modify the Simulink model, adding blocks that allow the perfect interface between control current and the grid.

The considered signals are those that allow the best confirmation of the characteristics of predictive control, and these are:

- **i_a, i_b, i_c :** currents on the 3 different phases a, b and c, for experimental tests measured with sensors in the cabinet.
- **v_{an}, v_{bn}, v_{cn} :** voltages measured directly on the load with sensors for the experimental tests.
- **i_{alfa} , i_{beta} :** currents obtained with transformation inside the Simulink model. For the experimental tests they are taken from the Control Desk.

An important result is to confront values of i_{alfa} measured and reference i_{alfa} during a big change in the value of the reference current, because this shows how the predictive control can quickly follow the reference value.

The last real important test is to look at the behaviour of the i_{alfa} and i_{beta} in a step condition. A real advantage of the predictive control with respect to other control methods like pulse width modulation (PWM), is the perfect decoupling between the 2 different signals. It is possible to see that a change in i_{alfa} does not have consequences on i_{beta} , and vice versa.

7.1. 1st experiment: DC link Voltage – Inverter – R Load

7.1.1. Test Parameters

Fig.7.1. shows the configuration used for simulations and experimental tests. The signals that have been measured and that are useful to verify the operation of the system, are also indicated.

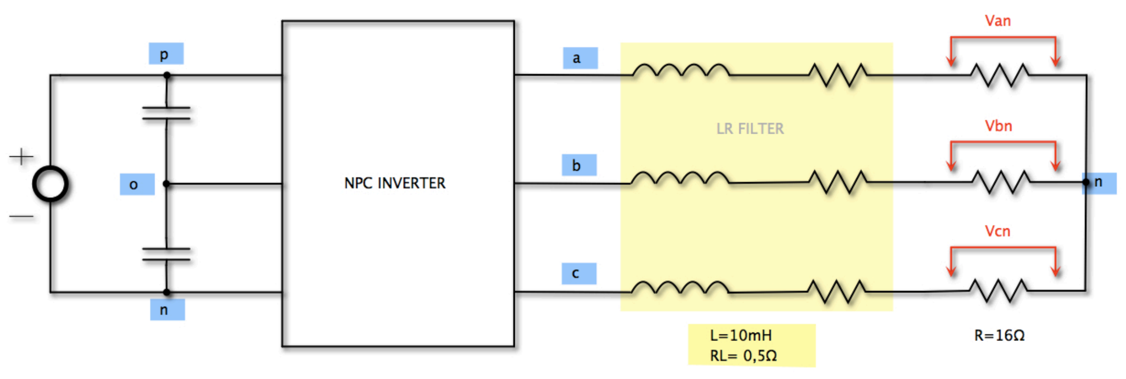


Fig. 7.1- Configuration for test nº1

Next tables shows the values of all parameters used in the simulations and in the laboratory experiments.

	Model Parameters
DC link voltage	180 V
DC link cap C1, C2	1,1 mF
R load	16 Ω
RL filter	0,5 Ω
L filter	10 mH

Tab. 7.1- Test Electrical Constants

	Values
Simulation Sampling Time	100 µs
Solver Simulink	ode1 (Euler)

Tab. 7.2- Simulation Parameters

7.1.2. Results

The waveforms for the relevant signals are shown.

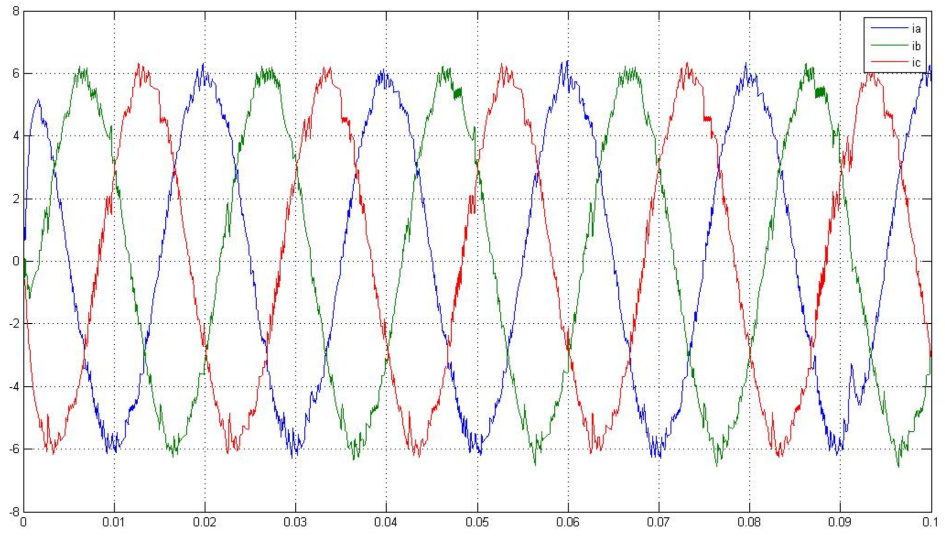


Fig. 7.2.: 3-phases currents i_a, i_b, i_c - Simulation

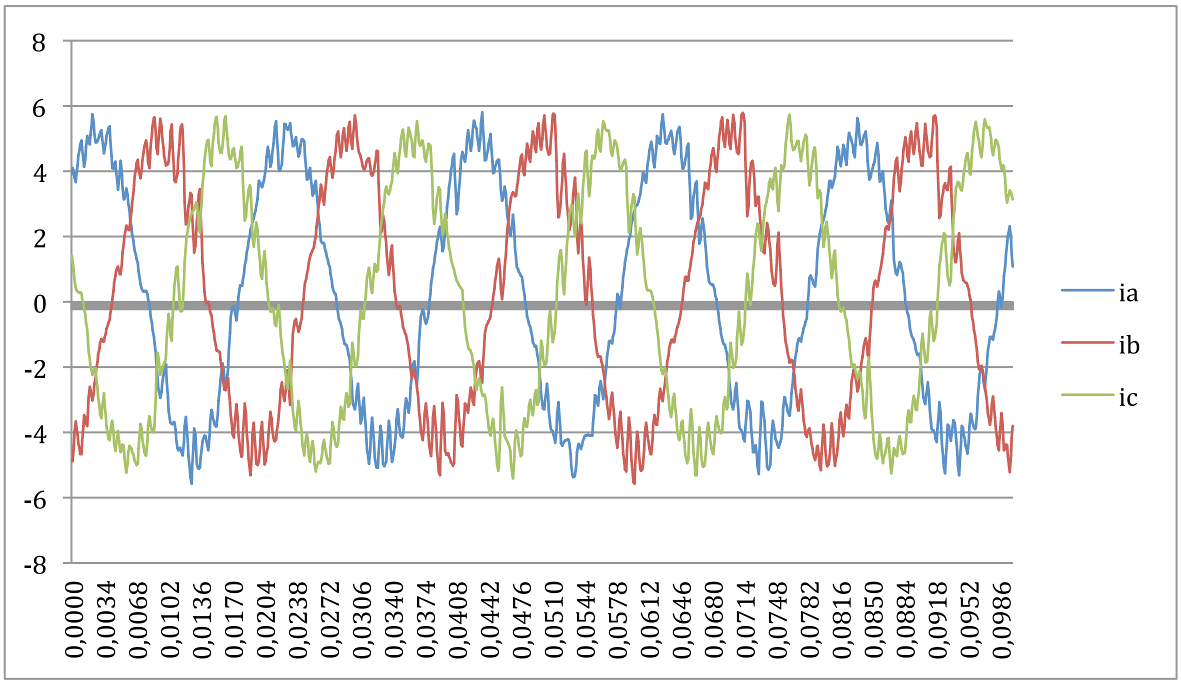


Fig. 7.3.: 3-phases currents i_a, i_b, i_c - Experimental

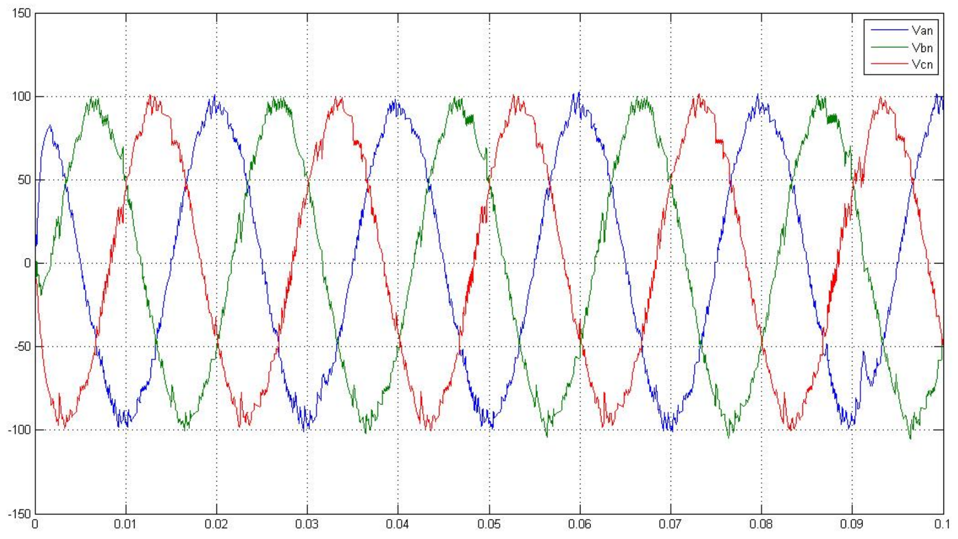


Fig. 7.4: 3-phases voltages Van, Vbn, Vcn - Simulation

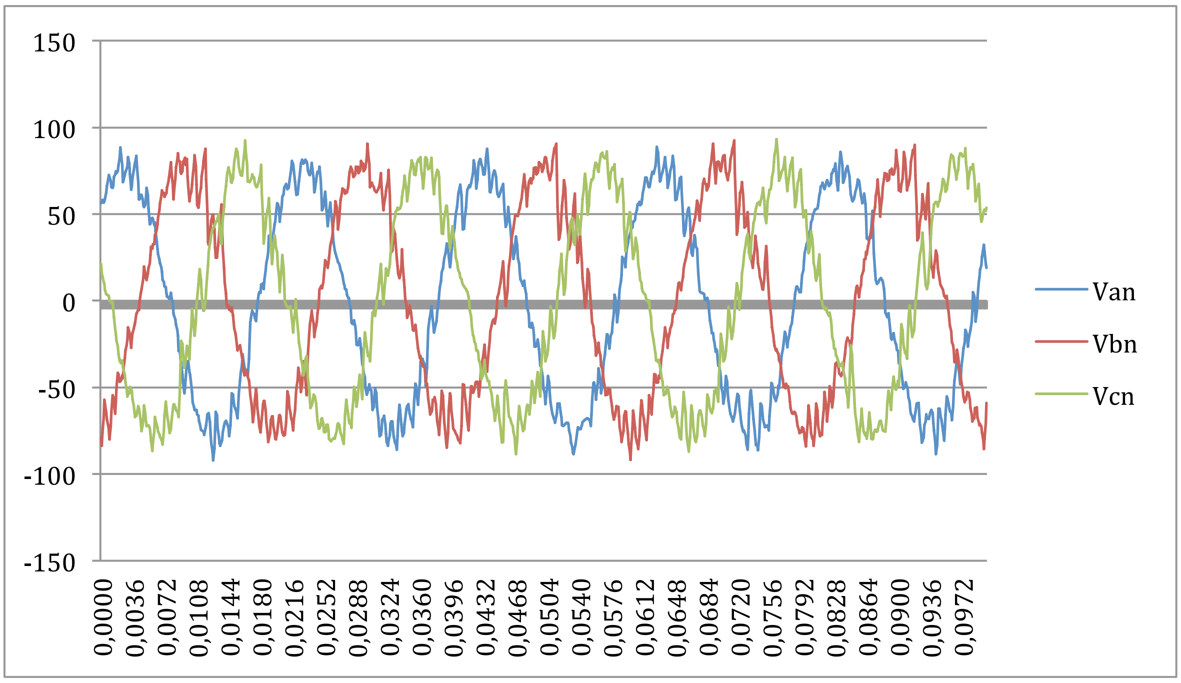


Fig. 7.5: 3-phases voltages Van, Vbn, Vcn - Experimental

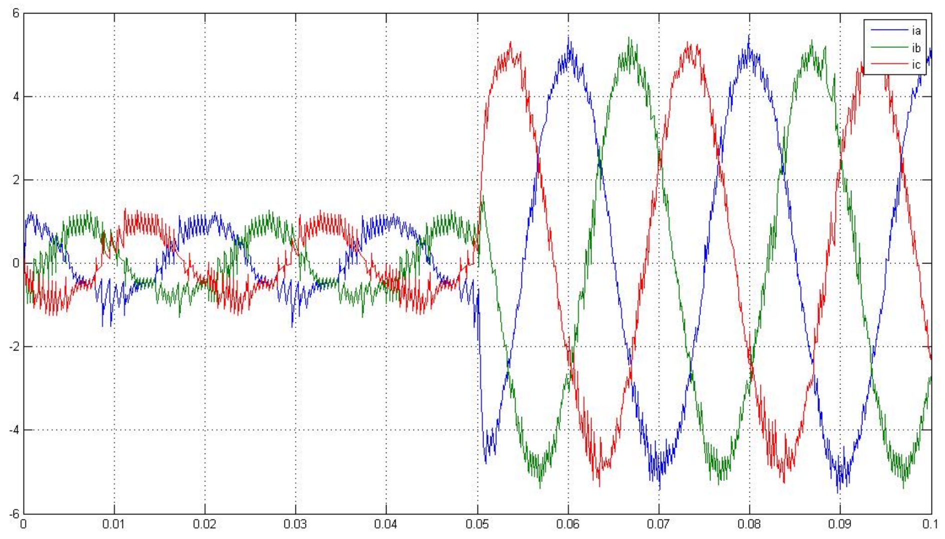


Fig. 7.6: Currents i_a, i_b, i_c with a 1-to-5 A step - Simulation

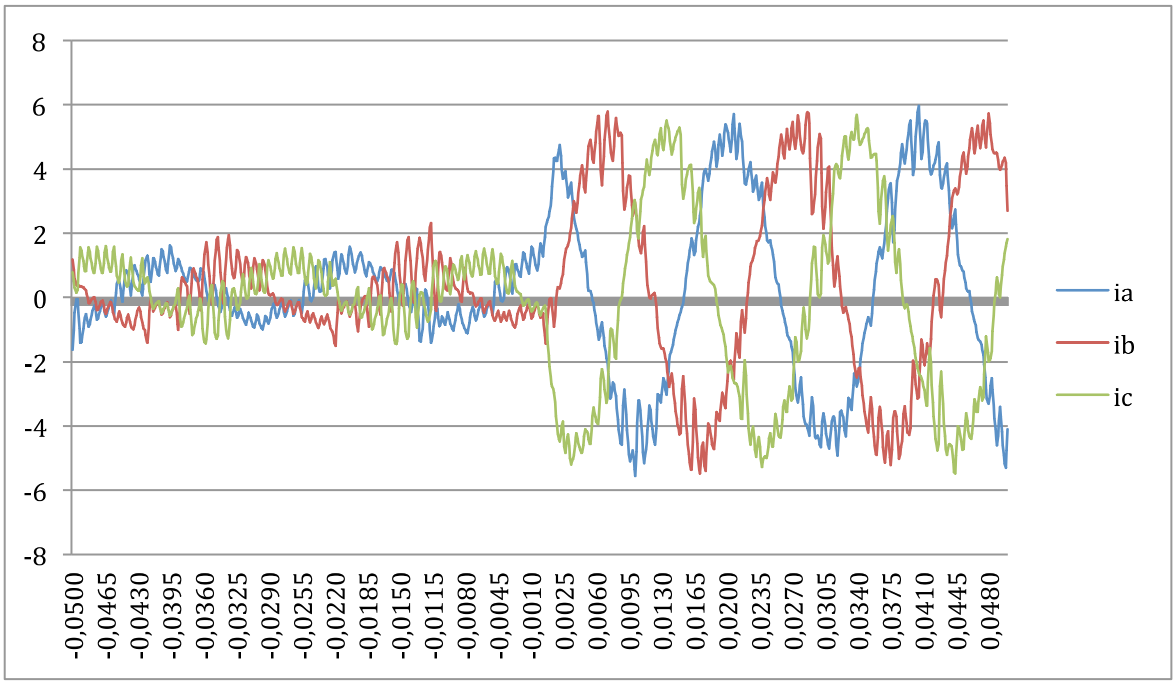


Fig. 7.7: Currents i_a, i_b, i_c with a 1-to-5 A step - Experimental

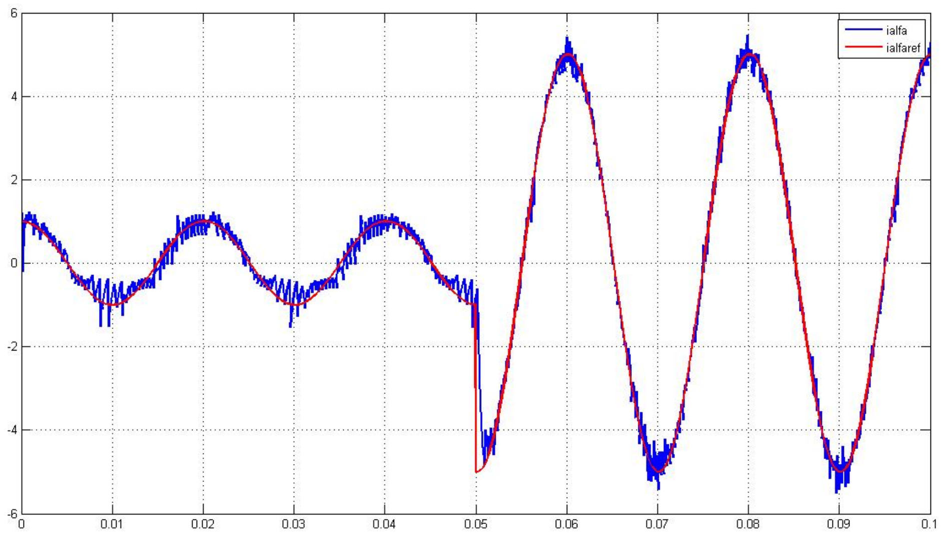


Fig. 7.8:Ialfa-ialfaref step 1 to 5 A with Rload - Simulation

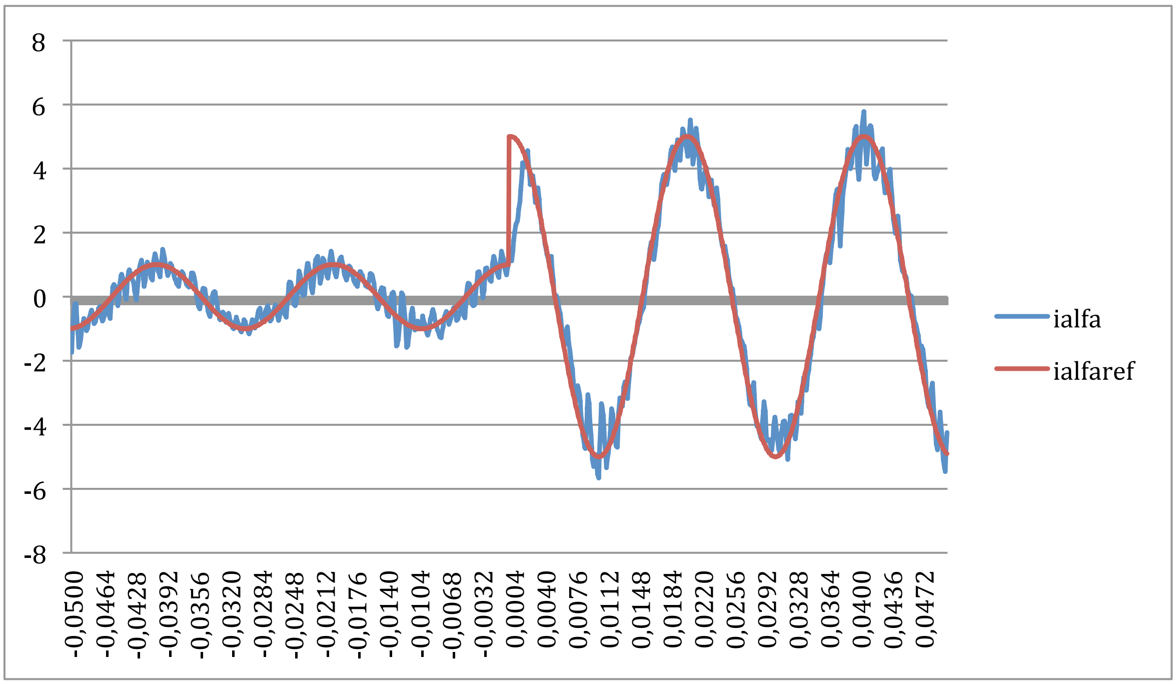


Fig. 7.9: Ialfa-ialfaref step 1 to 5 A with Rload – Experimental

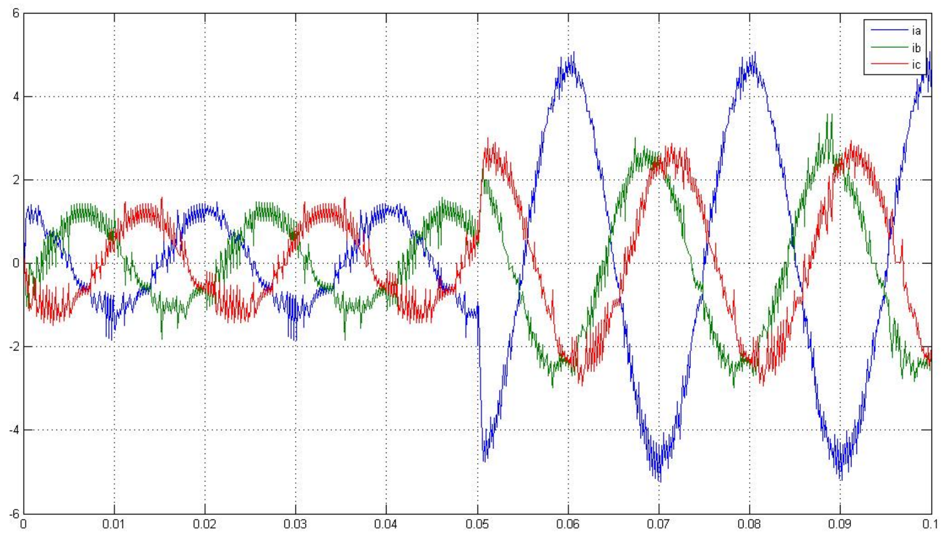


Fig. 7.10: 3-phases currents i_a, i_b, i_c step ialfa 1 to 5 – Simulation

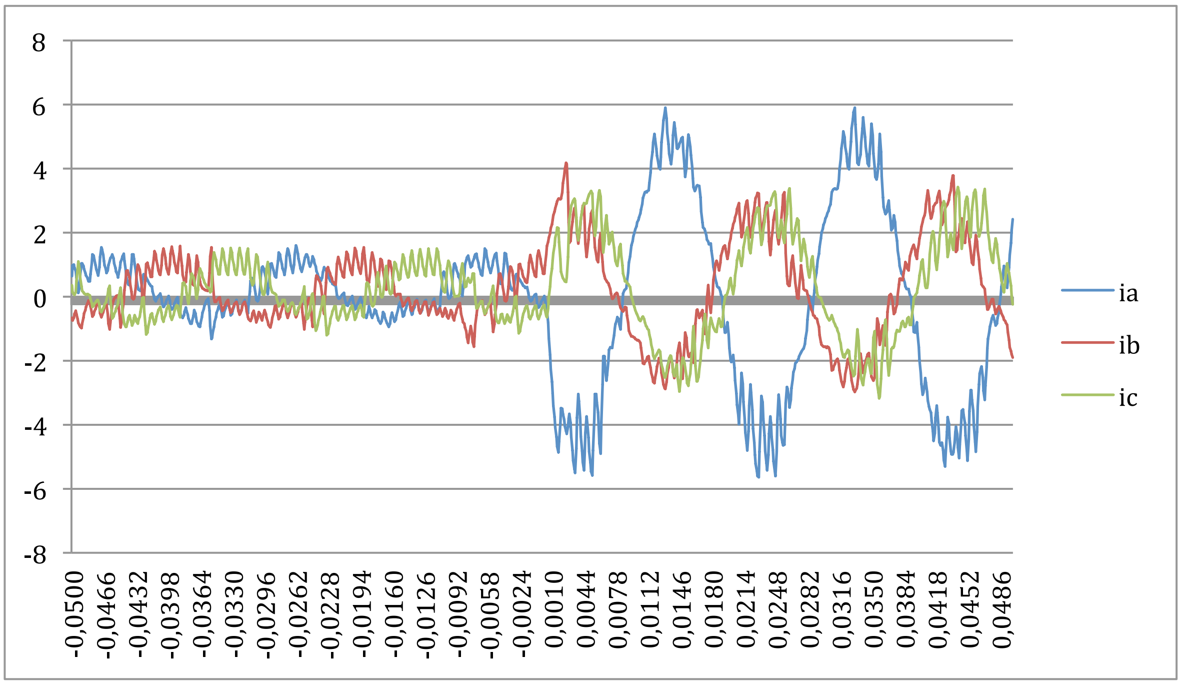


Fig. 7.11: 3-phases currents i_a, i_b, i_c step ialfa 1 to 5 – Experimental

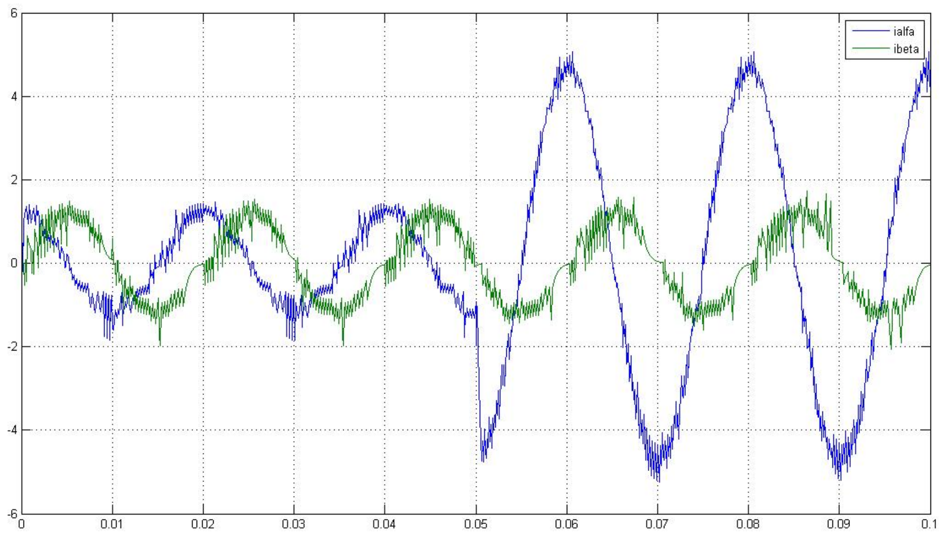


Fig. 7.12: ialfa-ibeta step of ialfa 1 to 5 A – Simulation

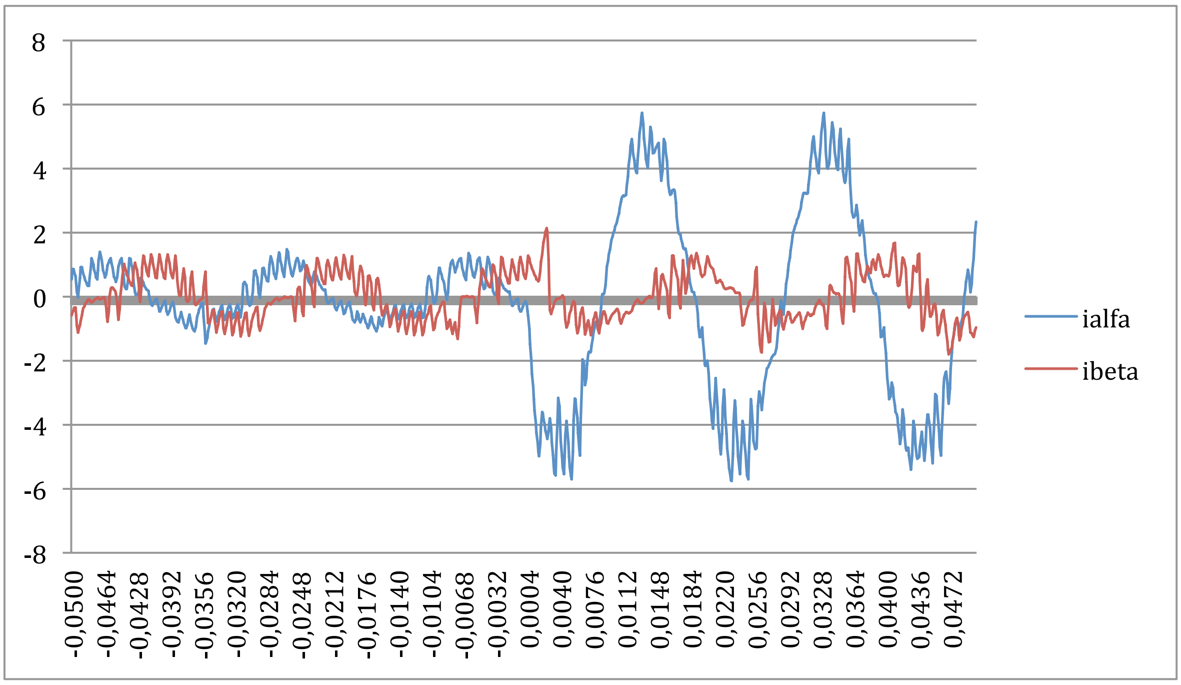


Fig. 7.13: ialfa-ibeta step of ialfa 1 to 5 A – Experimental

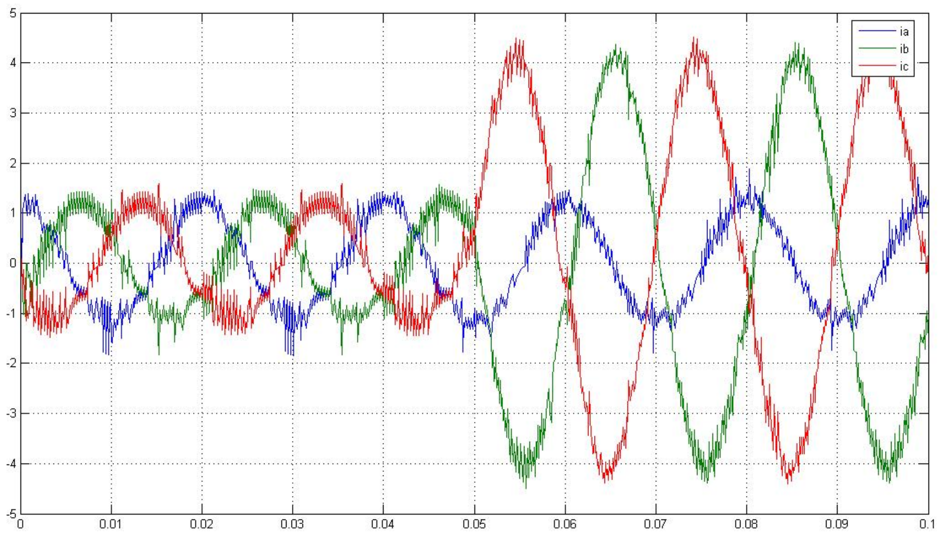


Fig. 7.14: 3-phases currents i_a, i_b, i_c step ibeta 1 to 5 – Simulation

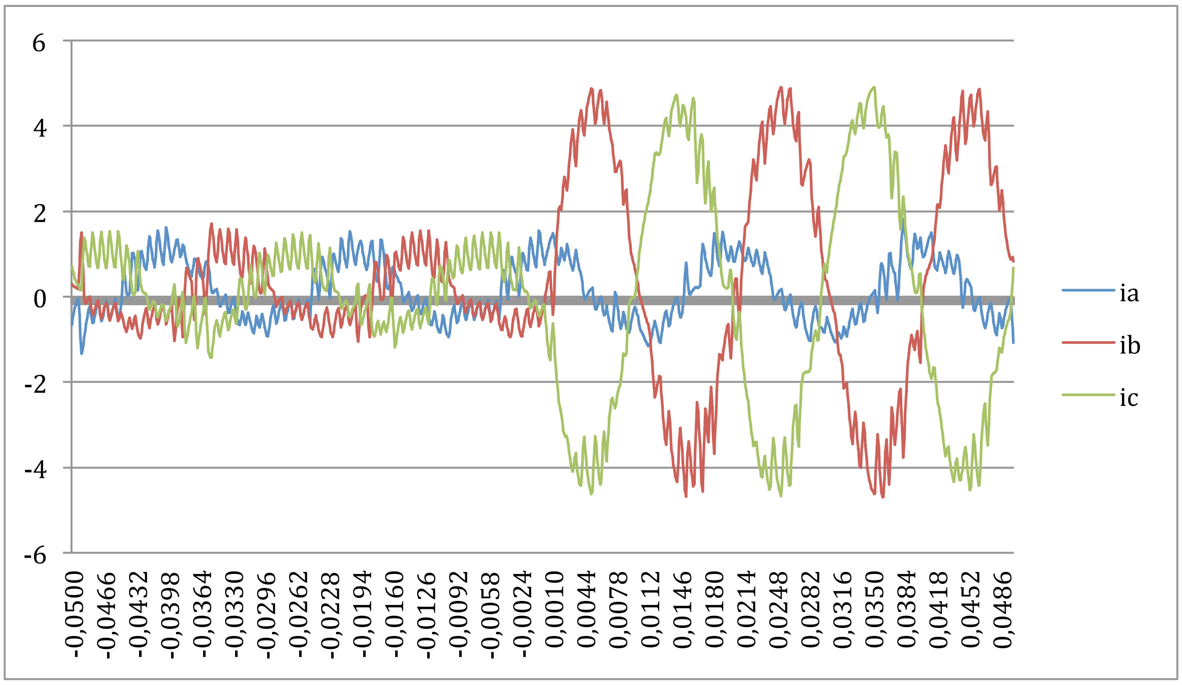


Fig. 7.15: 3-phases currents i_a, i_b, i_c step ibeta 1 to 5 – Experimental

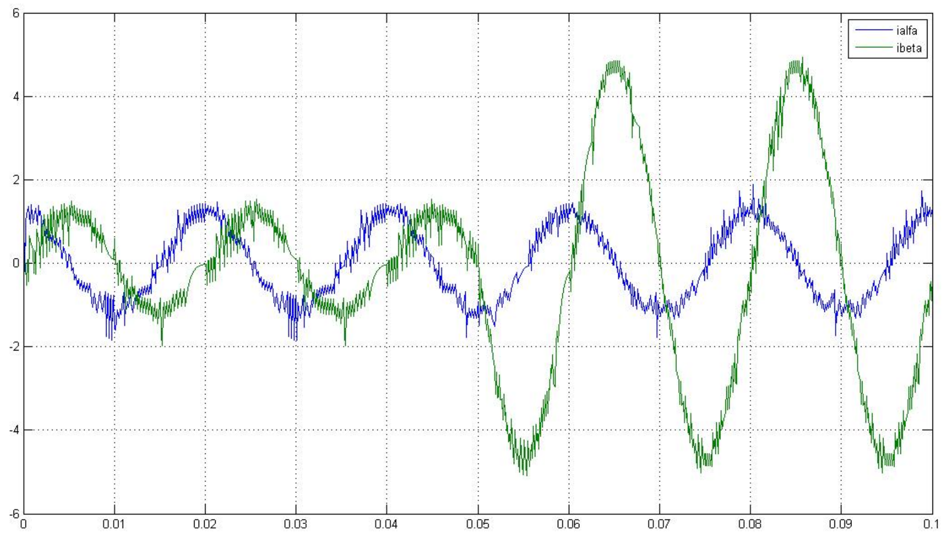


Fig. 7.16: ialfa-ibeta step of ibeta 1 to 5 A – Simulation

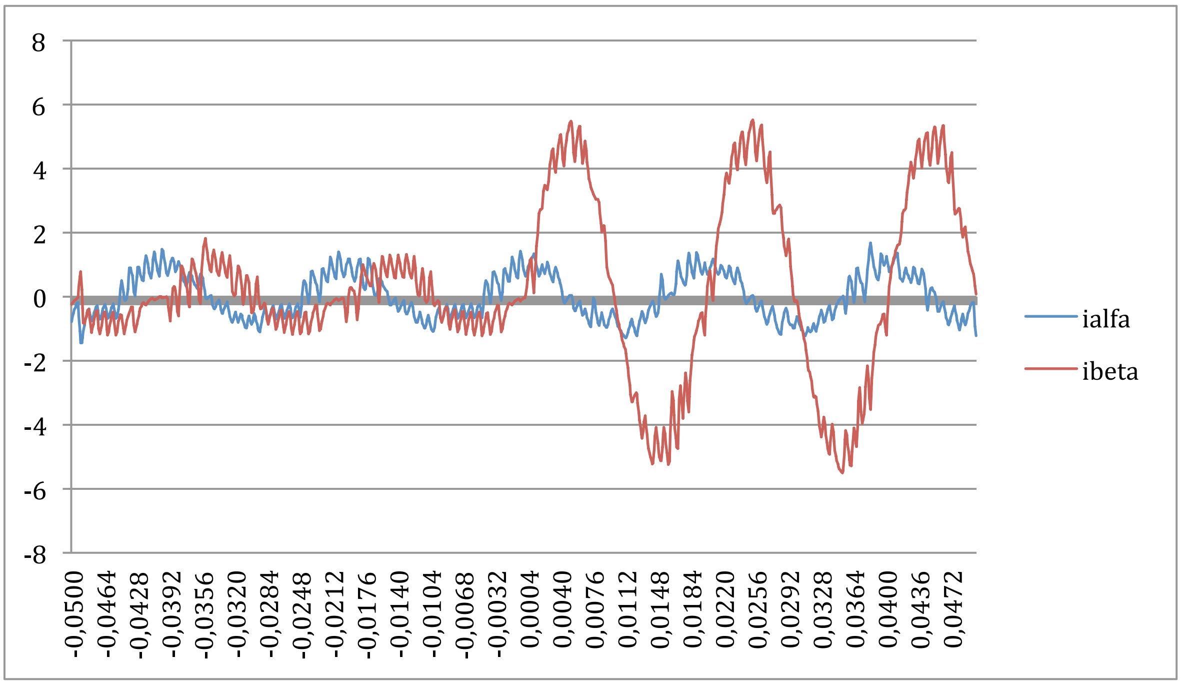


Fig. 7.17: ialfa-ibeta step of ibeta 1 to 5 A – Experimental

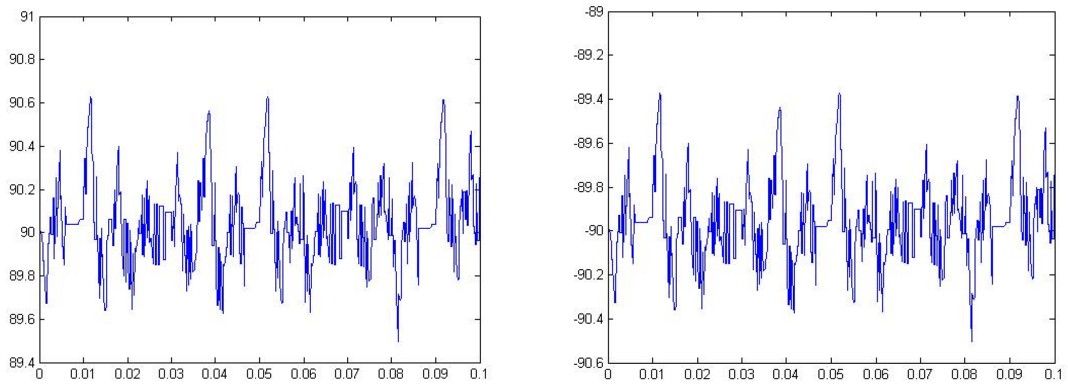


Fig. 7.18: Voltages Vp and Vn respectively – Simulation

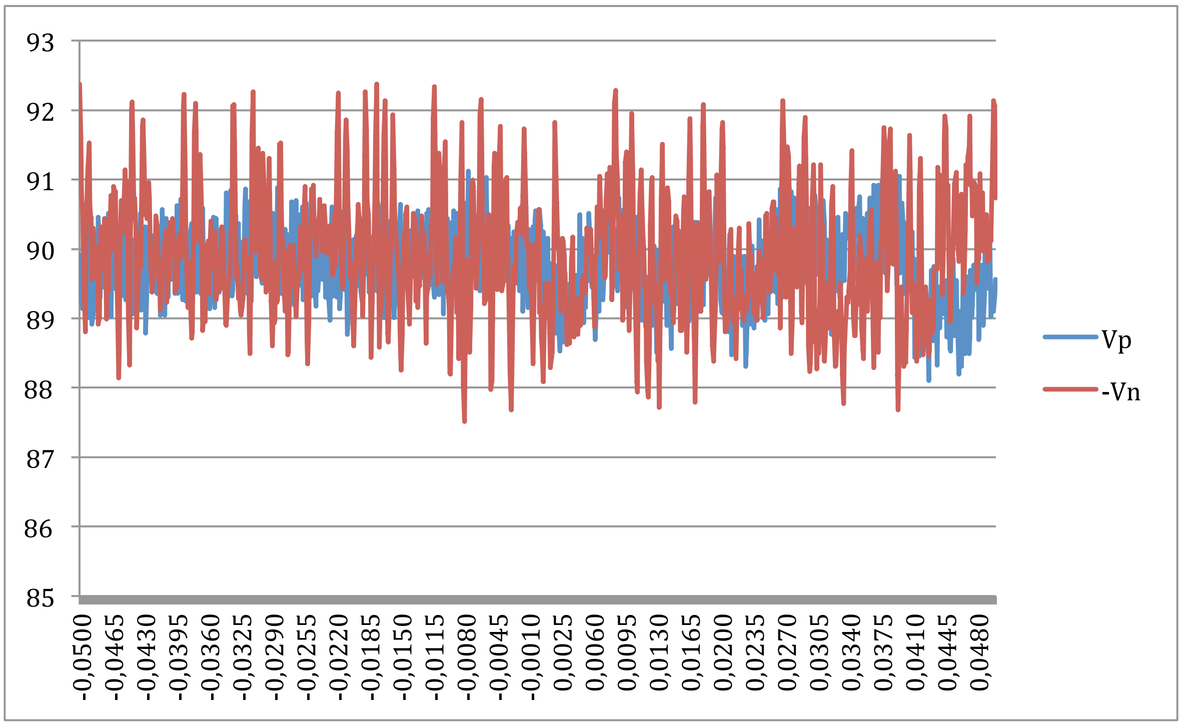


Fig. 7.19: Voltages Vp and (-Vn) - Experimental

7.2. 2nd experiment: DC link Voltage – Inverter – Grid

7.2.1. Test Parameters

Fig.7.1. shows the configuration used for simulations and experimental tests. The signals that have been measured and that are useful to verify the operation of the system, are also indicated.

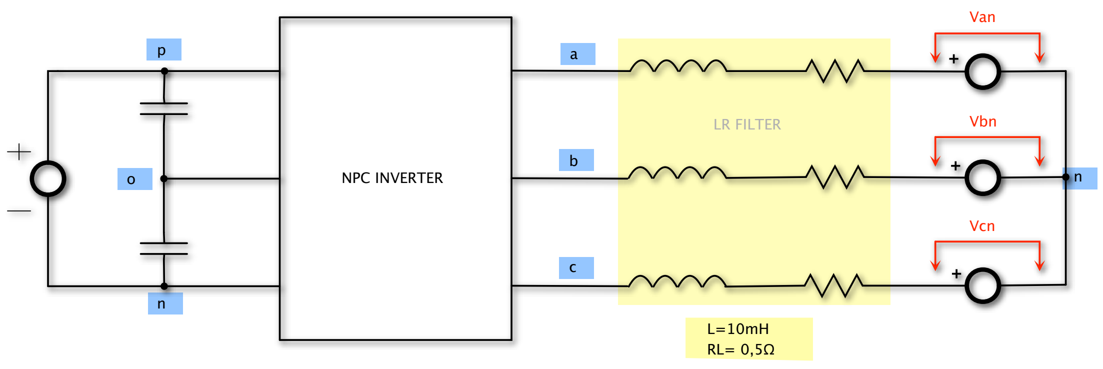


Fig. 7.20: Configuration for test nº2

	Model Parameters
DC link voltage	180 V
DC link cap C1, C2	1,1 mF
Grid Voltage	50 V
RL filter	0,5 Ω
L filter	10 mH

Tab. 7.3- Test Electrical Constants

	Values
Simulation Sampling Time	100 µs
Solver Simulink	ode1 (Euler)

Tab. 7.4: Simulation Parameters

7.2.2. Results

The waveforms for the relevant signals are shown.

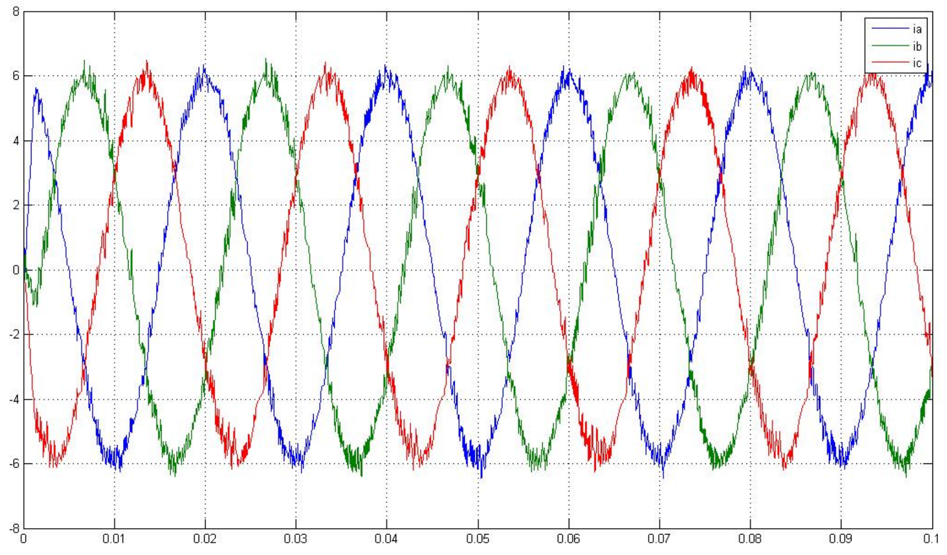


Fig. 7.21: 3-phases currents i_a, i_b, i_c - Simulation

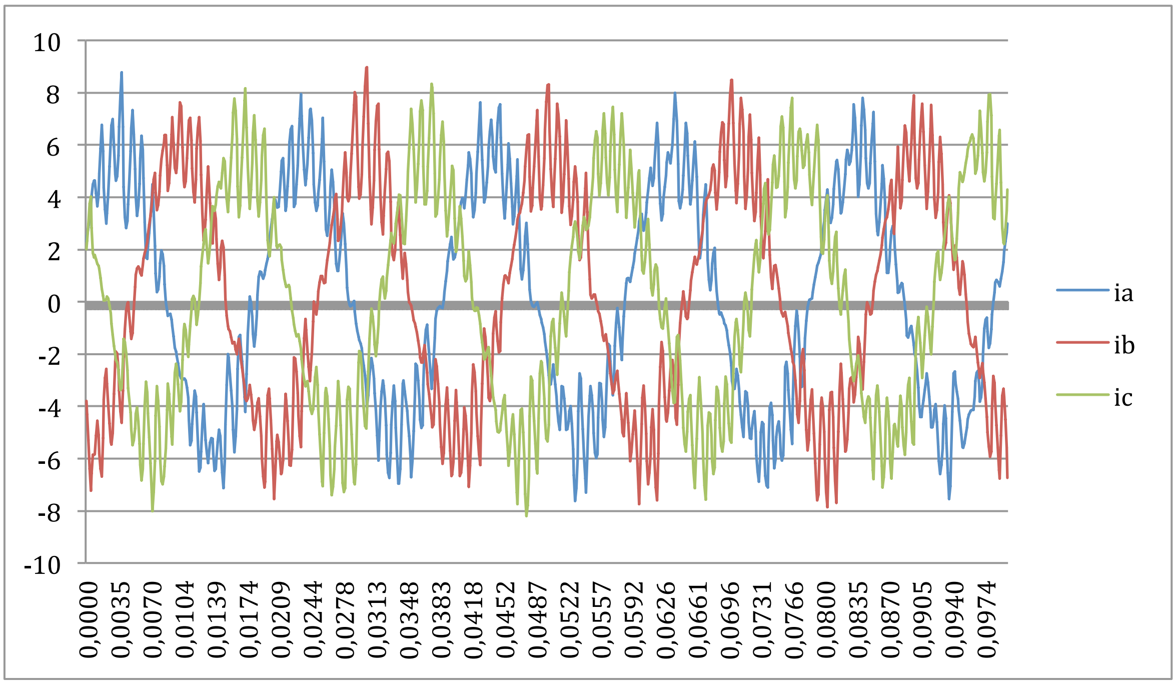


Fig. 7.22: 3-phases currents i_a, i_b, i_c - Experimental

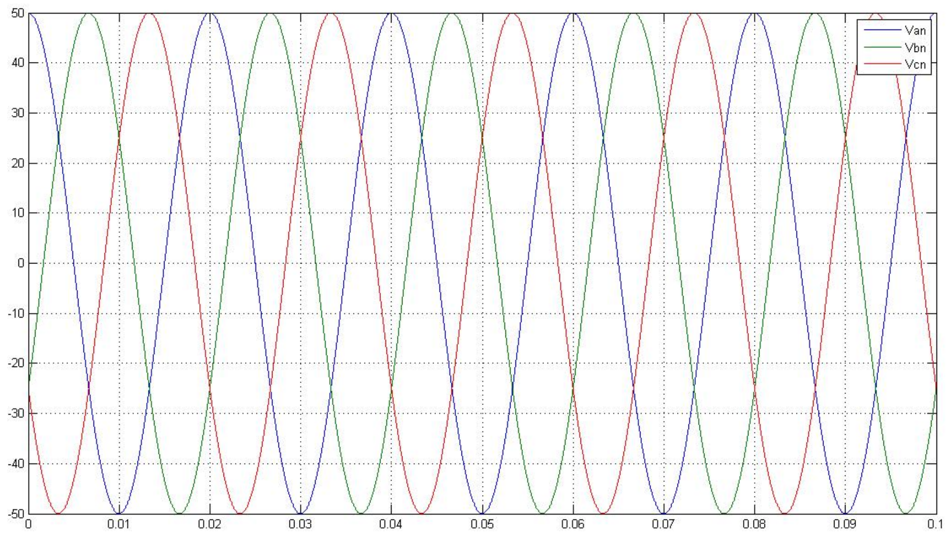


Fig. 7.23: 3-phases voltages Van, Vbn, Vcn - Simulation

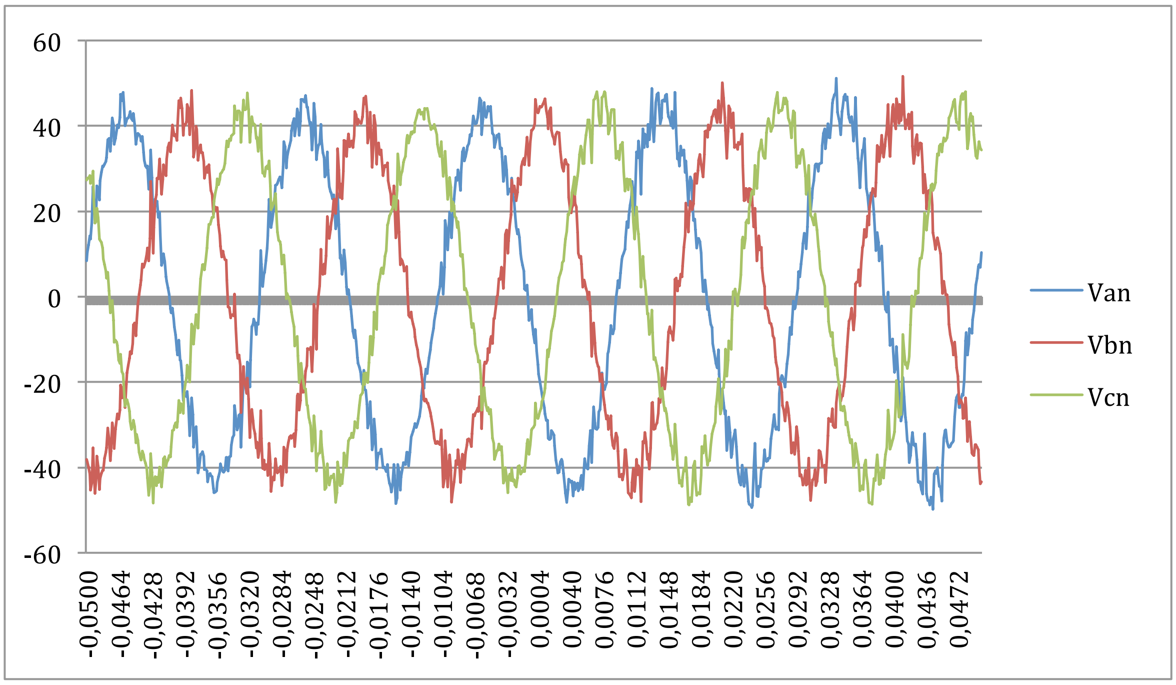


Fig. 7.24: 3-phases voltages Van, Vbn, Vcn - Experimental

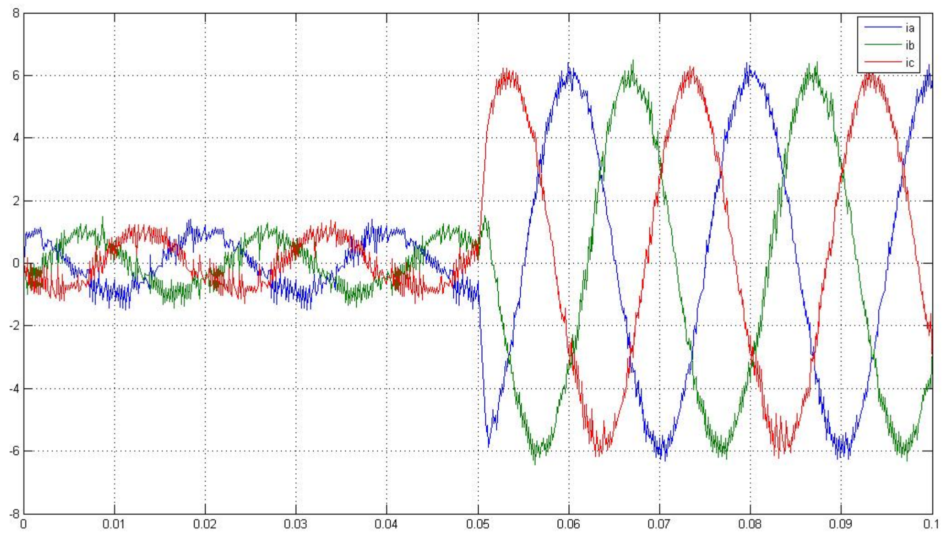


Fig. 7.25: iabc step 1 to 6 grid load - Simulation

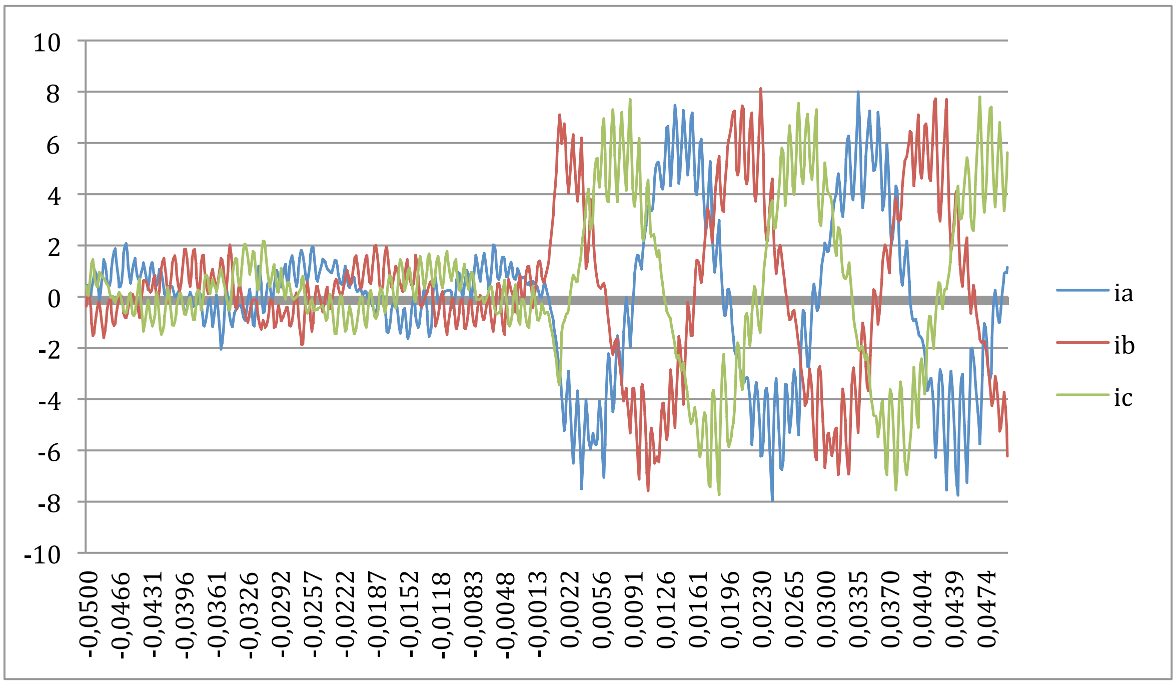


Fig. 7.26: iabc step 1 to 6 grid load - Experimental

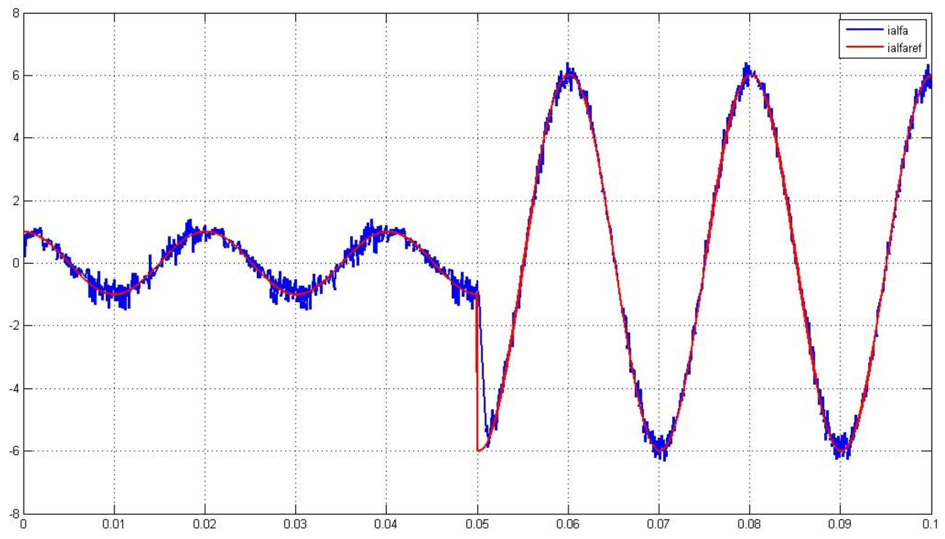


Fig. 7.27: ialfa-ialfaref step 1 to 6 A with grid load - Simulation

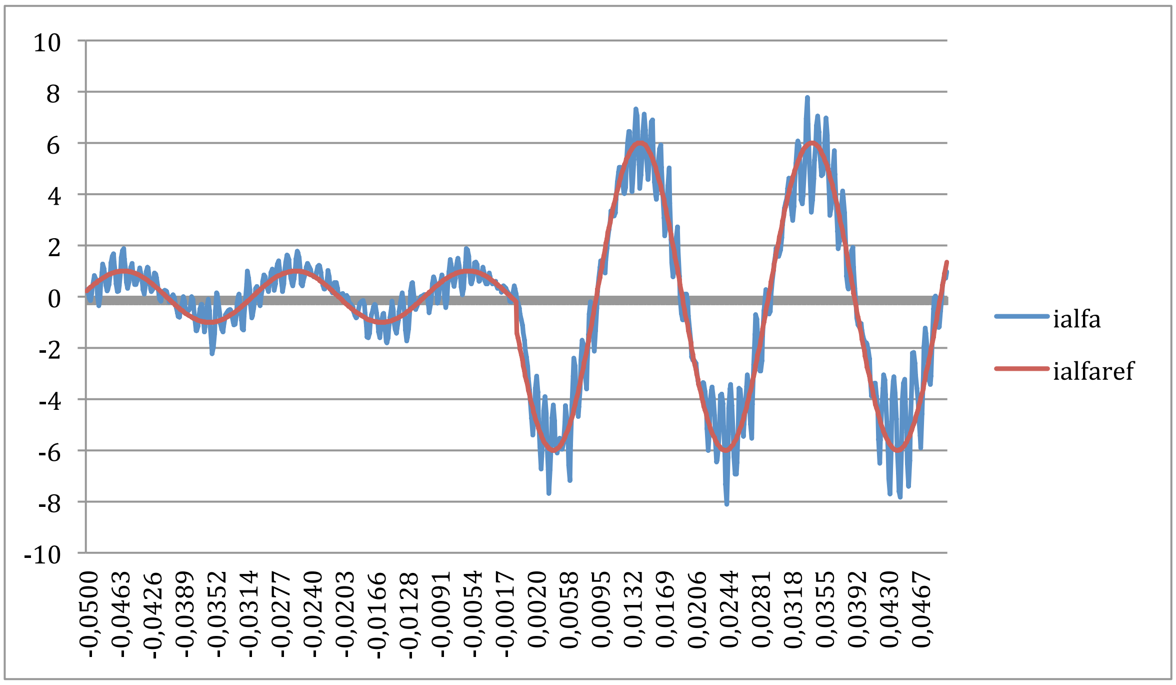


Fig. 7.28: ialfa-ialfaref step 1 to 6 A with grid load - Experimental

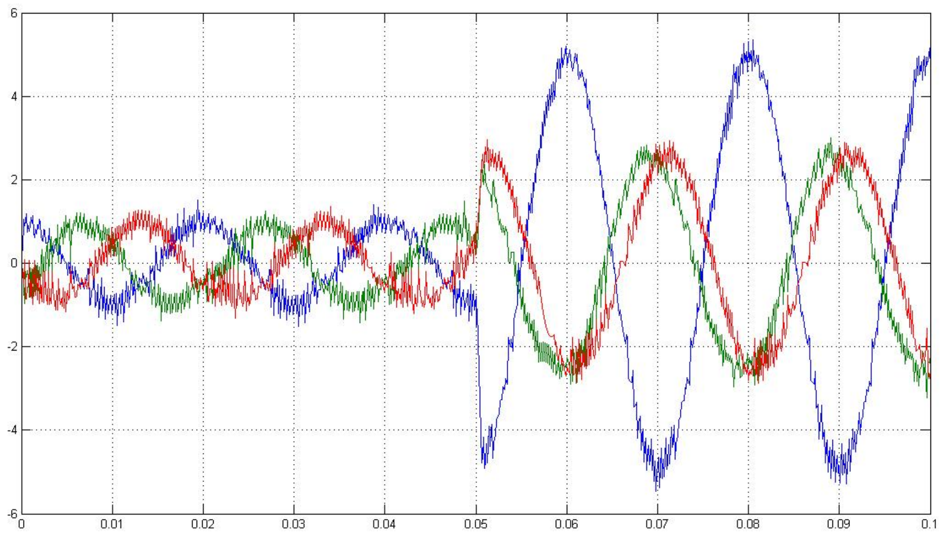


Fig. 7.29: 3-phases currents i_a, i_b, i_c step ialfa 1 to 5 – Simulation

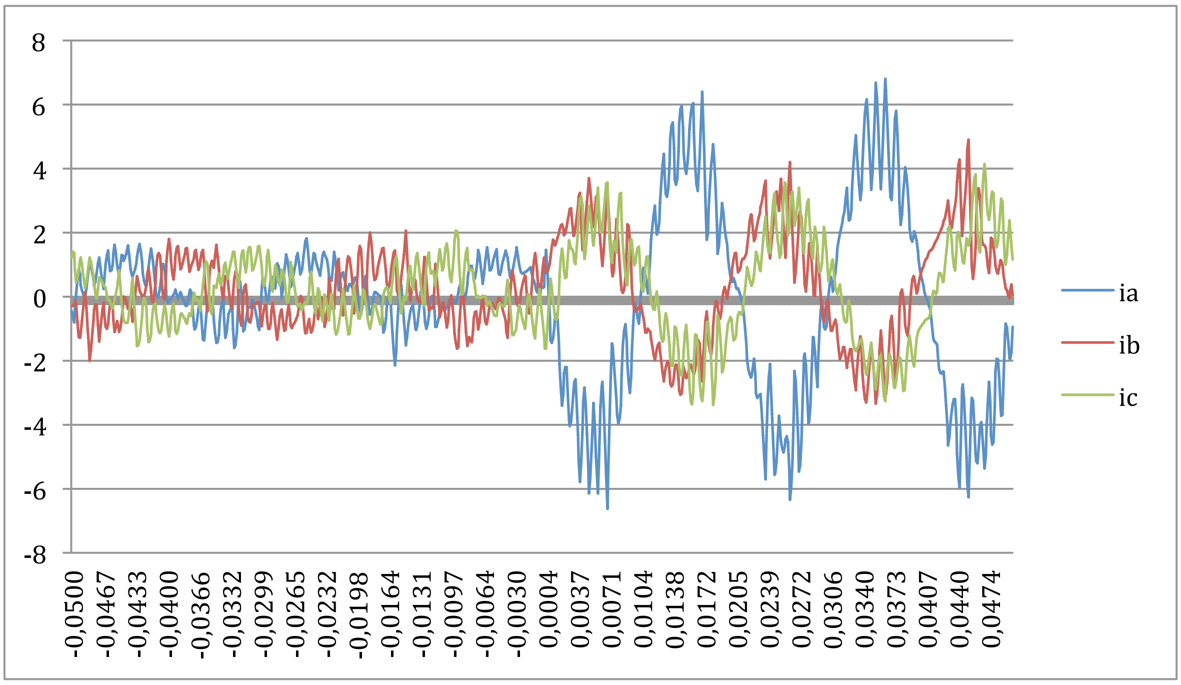


Fig. 7.30: 3-phases currents i_a, i_b, i_c step ialfa 1 to 5 – Experimental

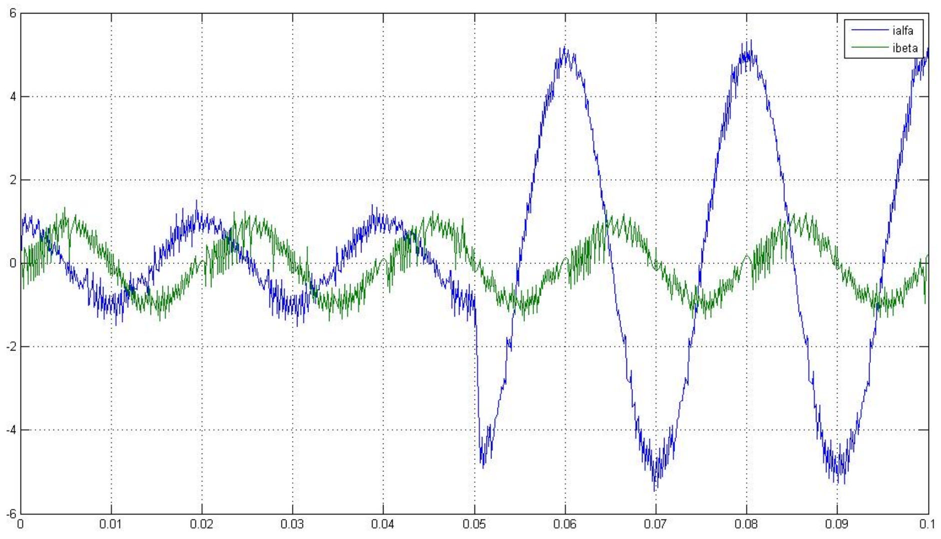


Fig. 7.31: ialfa-ibeta step of ialfa 1 to 5 A – Simulation

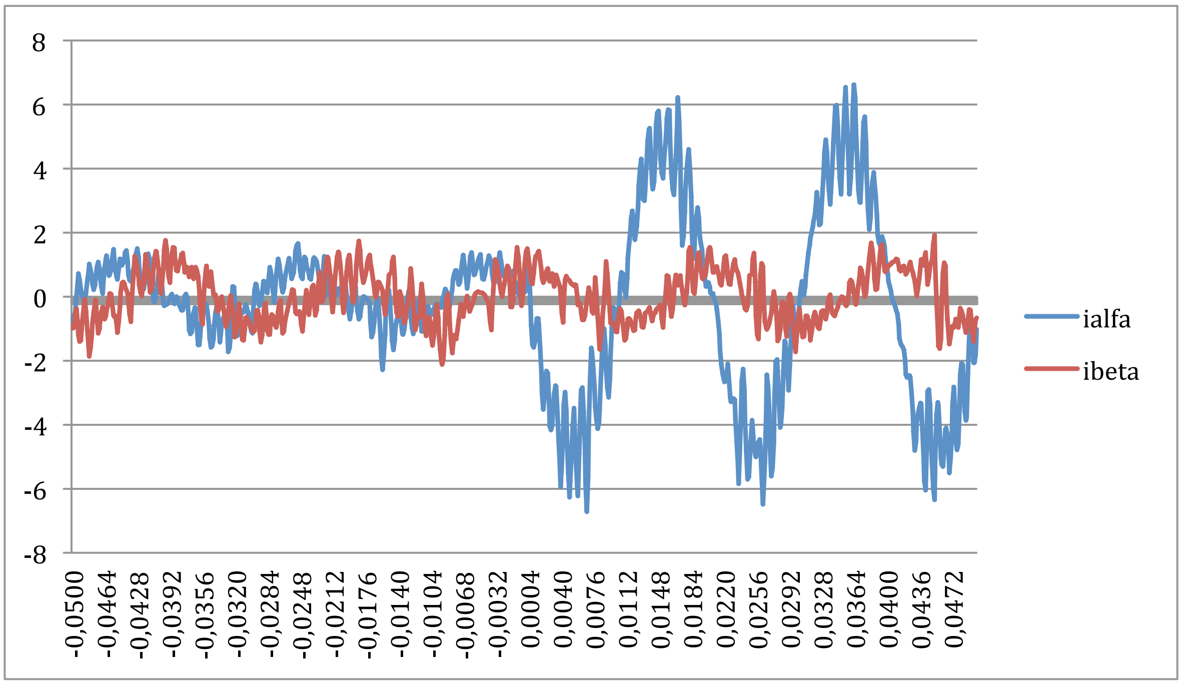


Fig. 7.32: ialfa-ibeta step of ialfa 1 to 5 A – Experimental

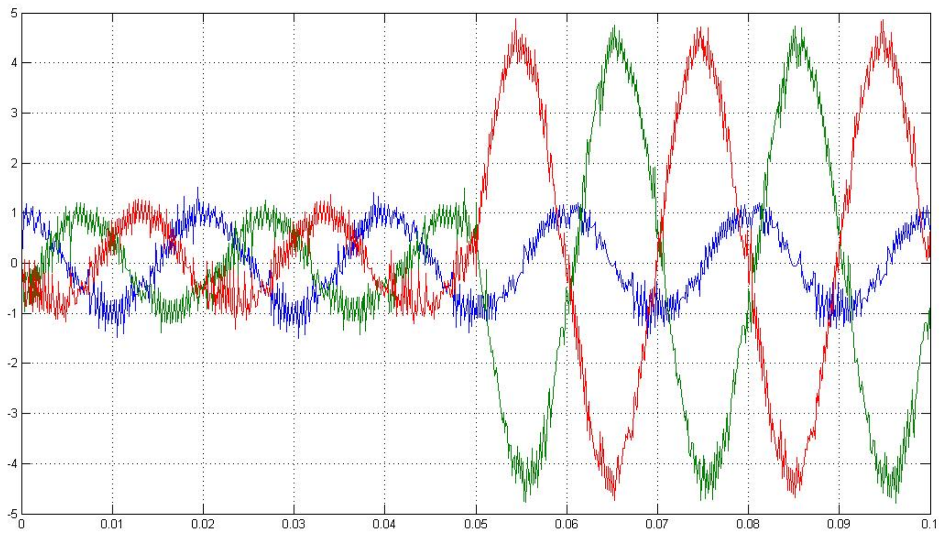


Fig. 7.33: ialfa-ibeta step of ibeta 1 to 5 A – Simulation

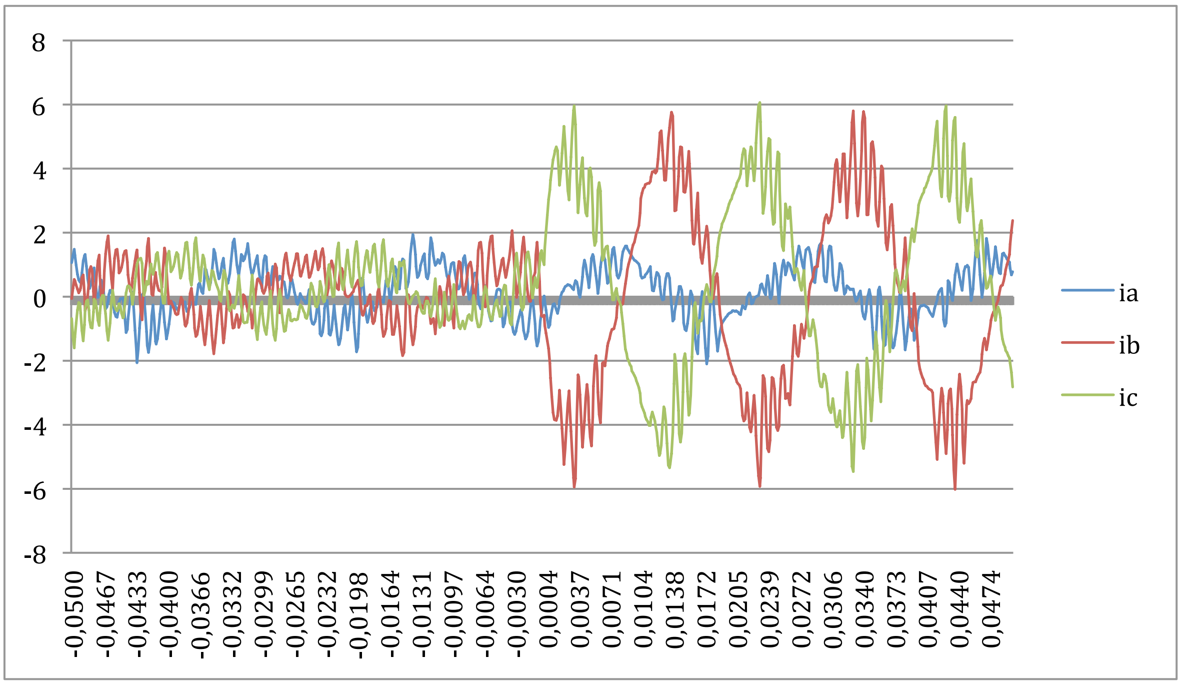


Fig. 7.34: ialfa-ibeta step of ibeta 1 to 5 A – Experimental

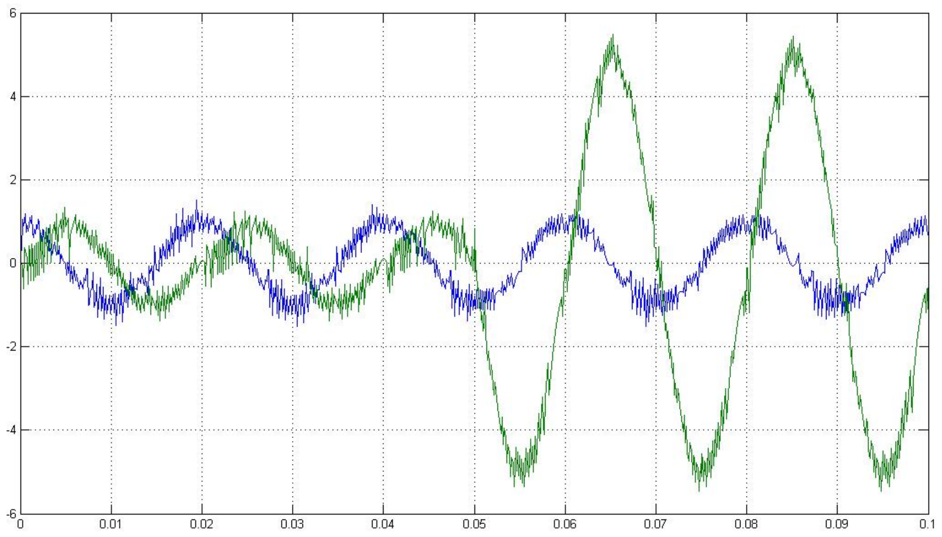


Fig. 7.35: ialfa-ibeta step of ibeta 1 to 5 A – Simulation

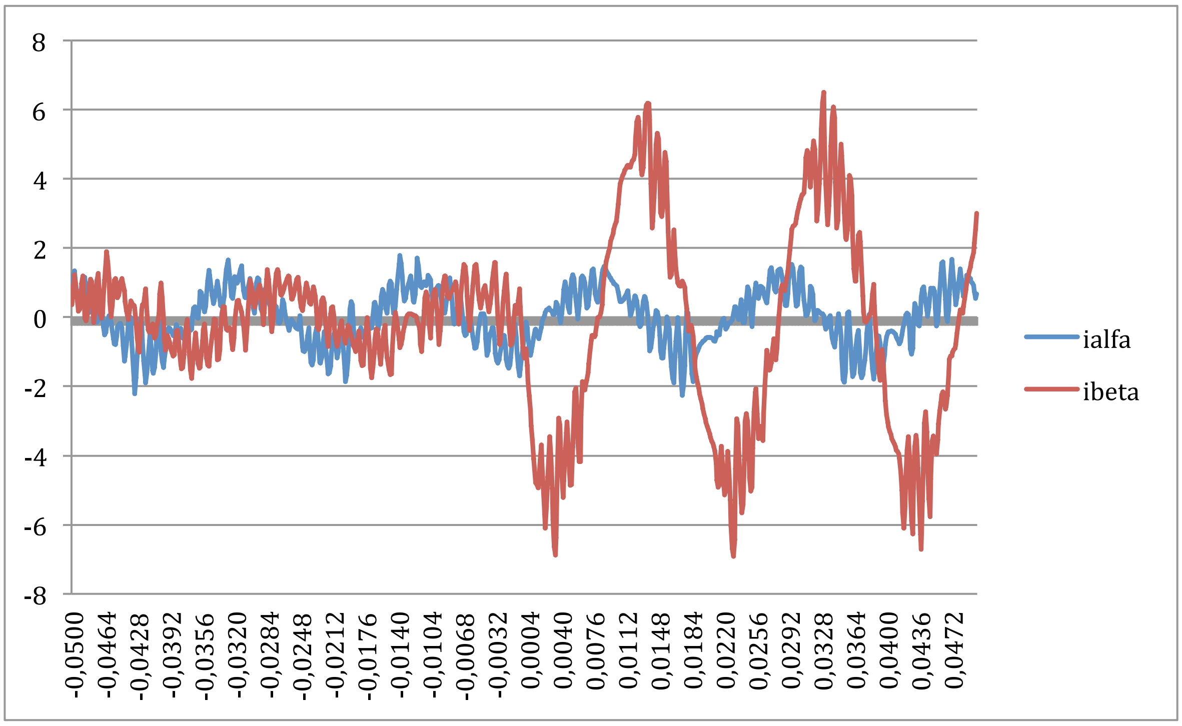


Fig. 7.36: ialfa-ibeta step of ibeta 1 to 5 A – Experimental

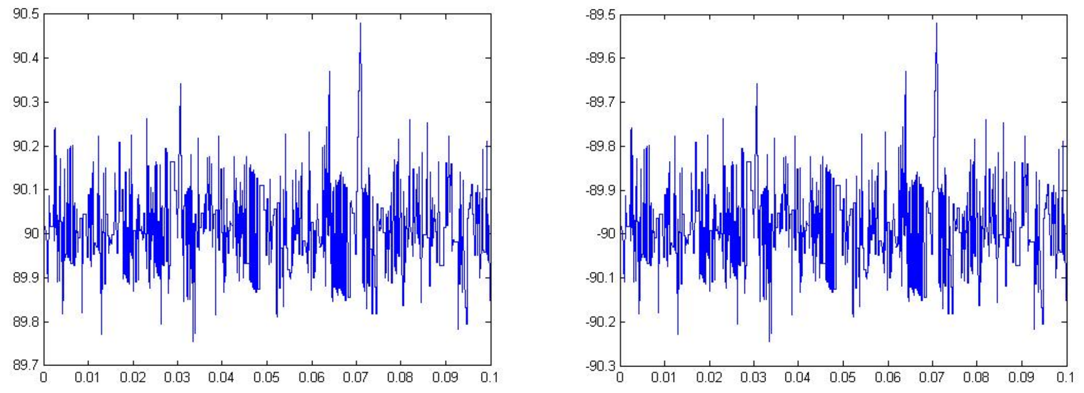


Fig. 7.37: Voltages Vp and Vn respectively – Simulation

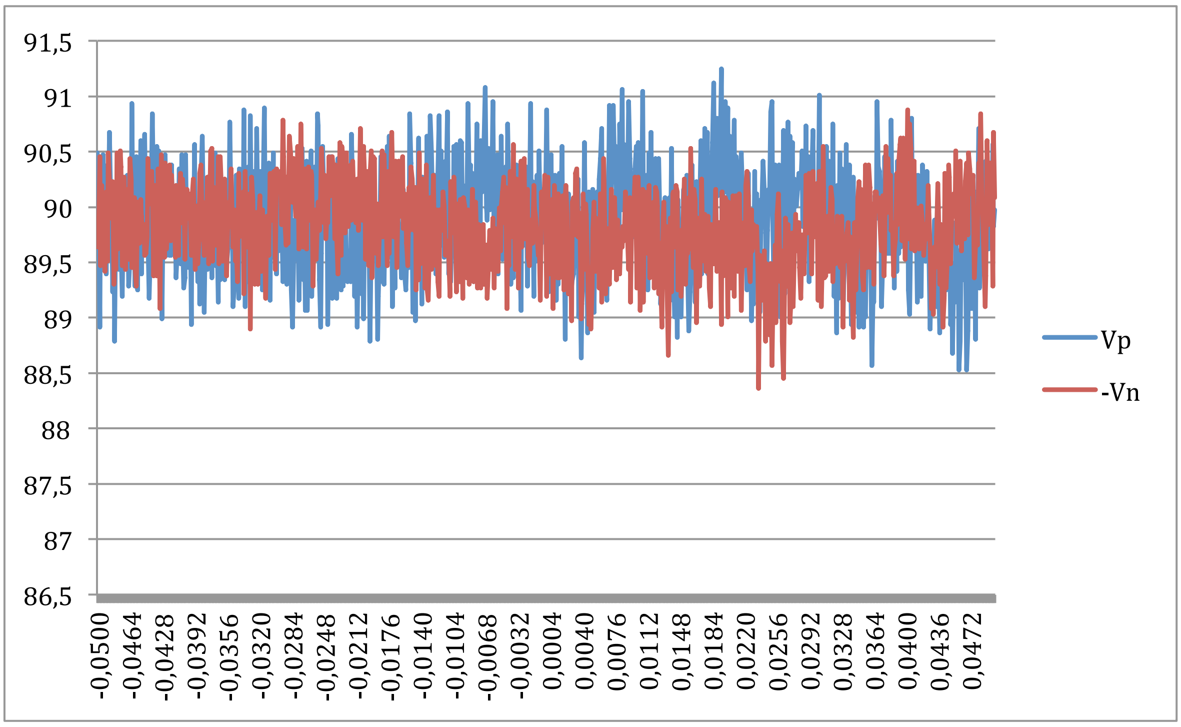


Fig. 7.38: Voltages Vp and (-Vn) - Experimental

7.3. Comments

Tests have verified the correct functioning of the system, with contained differences between simulations and experimental examination.

Looking at the experimental currents, is possible to see curling values, with a delta of ±2 A, while with simulation this oscillation had lower values. This difference may depend on characteristics of hardware used, that are designed for 1kV applications! We reached at most 180 V and 6 amps. For sure we can see difference between 60 V and 180 V like in next figure, seeing a reduction in the importance of the variation. So it is possible to conclude that further increasing the voltage value of DC link can reduce the problem.

8. Conclusions

In this project the application of the Predictive Control to a three-phase NPC converter has been realized. The main conclusion, looking at the experimental results obtained, is that this kind of control works perfectly on the laboratory inverter, achieving main objectives.

Highlights of this project are:

- Realization of Matlab Simulink diagrams for the implementation of the Predictive Control on a three-phase NPC converter.
- Design of FPGA and dSpace programming for the application of the control to GREP laboratory’s equipment.
- Realization of a Simulink model for making simulation of the system before than directly testing it on the equipment.
- Realization of test of functioning of the FPGA programming, using an oscilloscope connected to digital connection board.
- Realization of an User Interface with Control Desk, for real time changes during experimental tests.
- Experimental tests of entire system, with acquisition of all useful data from the laboratory equipment.
- Comparison of simulation and experimental results, obtaining satisfactory outcomes of the tests.
- Achievement of a further test, by connecting the output of the system to the grid and verifying proper operation of the system even in this condition.

As points to improve the system in the future we may mention:

- Realize the entire control with a reduction in the execution time. Possibly an entire C language programming of the dSpace instead of Simulink can achieve this objective.

- Implement the entire system using recent equipment, with goal of increment the complexity of the system and without problems in execution time.

- Realize a predictive control with 2 steps prediction of the current. In this way in every sampling period the value of the prediction of 2 future's sampling period are always available, with better performance of the system.

- Test the system in an entire wind plant emulator. It will be necessary to develop of the predictive control of the rectifier stage to be coupled to the system analysed in this project.

Realization of this project gives to the student next chances:

- The functioning of Matlab platform has been investigated, with improvements in the use of this environment.

- The knowledge of VHDL and FPGA functioning has improved, realizing the FPGA programming for the connection between dSpace and converter.

- Has been improved the knowledge of the dSpace system ha improved, as well as all the programs necessary to its use, like Simulink and Control Desk.

- The topology of a 3-phase NPC converter has been studied for first time, knowing advantages of this kind of inverter in high power installations.

9. Acknowledgements

I want to thank very much my project directors Josep Bordonau and Alejandro Calle, especially the second one, who gave me an invaluable aid in the realization of this project.

I want to thank the rest of the companions of GREP, Joan Nicholas, Sergi Busquets and especially Àlber Filbà, who has been a giant source of information always really useful.

Infine vorrei ringraziare la mia famiglia, gli amici e soprattutto mia madre, che “a distanza” mi ha aiutato anche nei momenti più delicati a terminare questo progetto, e anche a chi da più “vicino” mi ha accompagnato in questo ultimo periodo dei miei studi.

Un ringraziamento anche alla prof. Elena Maria Valcher, che mi ha reso molto più semplice portare a termine il progetto TIME da me svolto negli ultimi 3 anni, frequentando l’Università di Padova e la UPC ETSEIB Barcelona.

Thanks a lot.

Muchas Gracias.

Moltes Gràcies.

Grazie mille a tutti.

10. Bibliography

References

[1] ORTEGA, J. D.; *Emulador experimental de un sistema aerogenerador con tecnología multinivel*; Barcelona, Projecte Final de Carrera ETSEIB, 2005.

[2] CALLE, A.; *Emulador d'un sistema eòlic connectat a la xarxa elèctrica amb tecnologia back-to-back de tres nivells*; Barcelona, Projecte Final de Carrera ETSEIB, 2009.

[3] FILBÀ, A.; *Desenvolupament de la modulació SHE en un inversor NPC de 3 nivells;* Barcelona, Projecte Final de Carrera ETSEIB, 2011.

[4] RODRÍGUEZ, J., PONTT, J., SILVA, C.; *Predictive Current Control of a Voltage Source Inverter;* Proc. IEEE, vol.54, no.1, pp.495-503, feb. 2007.

[5] RODRÍGUEZ, J., PONTT, J., SILVA, C.; *Predictive Control of a Three-Phase Neutral-Point-Clamped Inverter;* Proc. IEEE, vol.54, no.5, pp.2697-2705, oct. 2007.

[6] A. Nabae, I. Takahashi, and H. Akagi, *A new neutral-point-clamped PWM invertir*;IEEE Trans. Ind. Appl., vol. IA-17, no. 5, pp. 518–523, Sep./Oct. 1981.

[7] Samir Kouro, Patricio Cortés, René Vargas, Ulrich Ammann, José Rodríguez; *Model Predictive Control—A Simple and Powerful Method to Control Power Converters;* IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL. 56, NO. 6, pp.1826-1838, jun. 2009.

Complementary Biography

HOLMES, D. GRAHAME, LIPO, THOMAS A.; *Pulse Width Modulation for Power Converters, Principles and practice;* Piscataway, NJ; IEEE Press Series on Power Engineering; p. 396–410.

RASHID, MUHAMMAD H.; *Power Electronics* Handbook; San Diego, California; Academic Press; p. 237

BOIX, ORIOL, *et. al.*; *Tecnología Eléctrica*; Barcelona, Ediciones CEYSA, 2002. AKIRA,

S. BUSQUETS-MONGE, S. SOMAVILLA, J. BORDONAU, D. BOROYEVICH, *A Novel Modulation for the Comprehensible Neutral-Point Balancing in the Three Level NPC Inverter with Minimum Output Switching-Frequency Ripple*, IEEE Power Electronics Specialists Conference, 2004.
DSPACE; *DS1103, Hardware Reference*; Alemanya; dSPACE 2003.

DSPACE; *RTIlib Reference*; Alemanya; dSPACE 2003.

DSPACE; *ControlDesk*, *Experiment Guide*; Alemanya; dSPACE 2001.

GILABERT, A., *Seminario dSPACE*, Barcelona, Grup de Recerca en Electrònica de Potència, Departament d'Enginyeria Electrònica; ETSEIB, 2003.
2001. 1998.

UNIVERSITY PROGRAM DESIGN LABORATORY PACKAGE; *User Guide*; Altera,

MATHWORKS, INC., *La edición de estudiante de Simulink*, Madrid, Prentice Hall,

MATHWORKS INC., THE; *Support*; [http://www.mathworks.com/support/]

11. dSpace Programming

The dSpace program was realized with Simulink graphical environment, included in Matlab suite. Was also necessary to added C language parts, included in the Simulink model like dedicated blocks. This has been necessary due to no compatibility between Matlab Code and Realtime Workshop included in this dSpace version.

C code has been integrated into Simulink diagrams generating S-Function blocks, after compiling the code in a DLL function, using the MATLAB MEX incorporated.

Names and type of I/O ports, as well as auxiliary functions necessary for code execution in the dSpace board, were defined inside the C code, using next template:

```
 /* I/O DEFINITION*/

static void mdlInitializeSizes(SimStruct *S)

 {

ssSetNumSFcnParams(S, 0);

if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount (S)) {

return;}

/* INPUTS */ /* N_Inputs = block inputs number!*/

if (!ssSetNumInputPorts(S, X)) return;

{ int_T i;

for (i=0; i< N_Inputs; i++) {

ssSetInputPortWidth(S, i, 1); ssSetInputPortDirectFeedThrough (S, i, 1);}}
```

```
/* OUTPUTS */ /* N_Outputs = block outputs number!*/

if (!ssSetNumOutputPorts(S, N_Outputs)) return;

{ int_T j;

for (j=0; j< N_Outputs; j++) {

ssSetOutputPortWidth(S,j,1);}

ssSetNumSampleTimes(S, 1);

ssSetOptions(S, SS_OPTION_EXCEPTION_FREE_CODE|
SS_OPTION_USE_TLC_WITH_ACCELERATOR |SS_OPTION_PLACE_ASAP); }

/* SAMPLE TIME INIT*/

static void mdlInitializeSampleTimes(SimStruct *S) {

ssSetSampleTime (S, 0, INHERITED_SAMPLE_TIME); ssSetOffsetTime (S, 0, 0.0);}

/* MAIN BLOCK START*/

static void mdlOutputs (SimStruct *S, int_T tid) {

/* ****** PROGRAM VARIABLES DEFINITION******* */

/* VARIABLES POINTERS SELECTIONS uPtr_i  FOR EVERY INPUT*/

/* i = input index, starting from 0 */

InputRealPtrsType uPtrs_i = ssGetInputPortRealSignalPtrs(S, i);

/* VARIABLES POINTERS SELECTIONS y_j  FOR EVERY OUTPUT */

/* j = output index, starting from 0 */

real_T *y_j = ssGetOutputPortRealSignal(S,j);



/* ****** MAIN CODE ******* */

}
```

```
/* ENDING */ static void mdlTerminate (SimStruct *S) { }

#ifdef MATLAB_MEX_FILE

#include "simulink.c"

#else

#include "cg_sfun.h"

#endif
```

Simulink Diagram

Next figure shows the entire scheme created in Simulink for experimental tests.

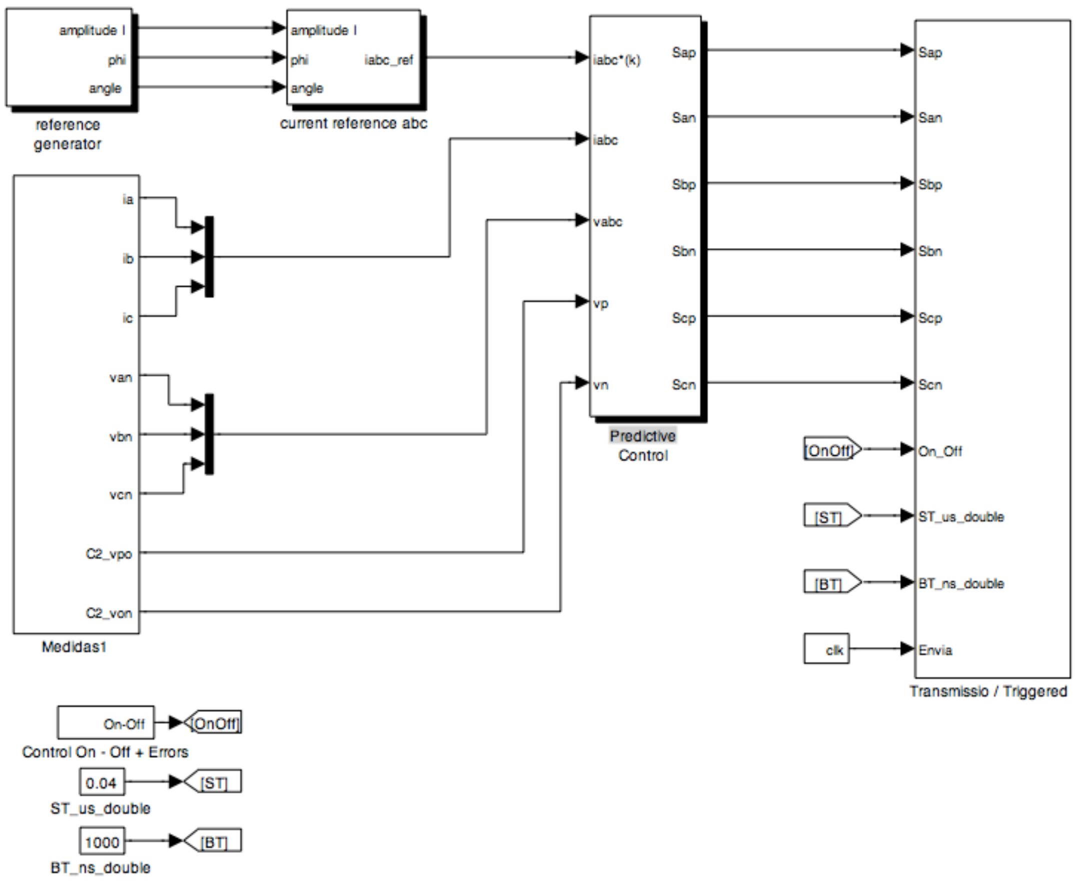


Fig. 11.1: General Simulink Scheme for dSpace programming

Now, all blocks that form the entire system are described.

11.1. Reference Generator

This block generates current, phase and angle references, used in the control for evaluate the differences with read values.

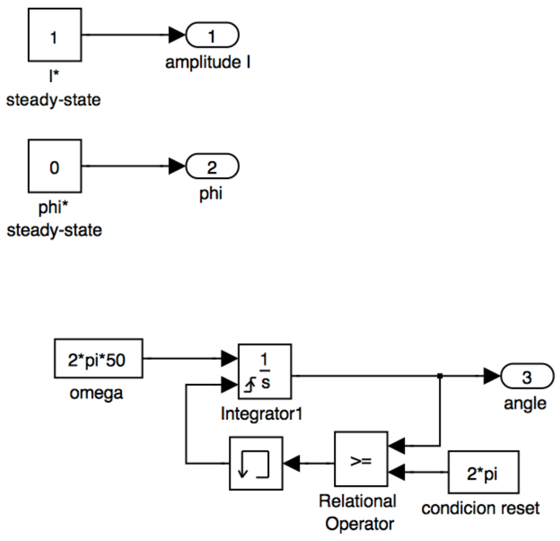


Fig. 11.2: Reference Generator block

amplitude I and *phi* are steady state values, that can be modified directly during the execution of tests from the Control Desk platform.

angle is a sawtooth wave generator for the angle reference, created in this way for reduce the calculation time respect to a generator block.

11.2. Current Reference abc

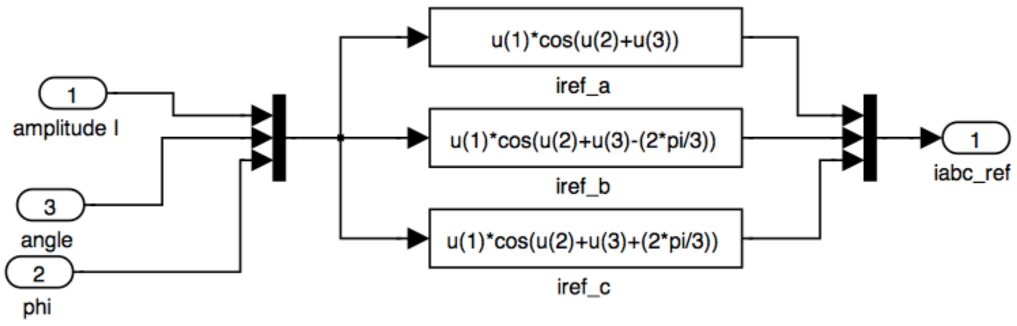


Fig. 11.3: Current Reference Block

This block create a 3-phase system of currents starting from amplitude, phase and anlge references. These currents are passed after to the Predictive Control block, for the effective realization of the control. The equations are:

$$i_a = amplitudeI \cdot \cos(phi + angle)$$
$$i_b = amplitudeI \cdot \cos(phi + angle - \frac{2\pi}{3})$$
$$i_c = amplitudeI \cdot \cos(phi + angle + \frac{2\pi}{3})$$

11.3. Medidas1

This block includes dSpace Simulink dedicated blocks, that allow to use the I/O of the board. The compiling directly recognized these blocks and activate the RTI lick, allowing the real time work of the entire system.

Fig. 11.4: Medidas1 Block

Three different ADC input are used, looking on next map of connections. The values of the offsets have been found in old projects, comparing PC read values with direct measure in the hardware with multimeters and oscilloscopes.

11.4. Transmissio/Triggered

Transmissió/Triggered block prepares the data and write it in the communication bus, allows the FPGA to read it. Next fig. shows the programming of this block

Was necessary to programming the transmission with C code for access to dSpace funcions. The execution of the C code inside the Simulink scheme is realized thank to an S-Function block, activated with the rising edge of the *Envia* input. This input is connected to *clk* block, what create a digital clock like described in 6.2.

Function ds1103_bit_io_write() is used for write data in the 32bit communication bus.

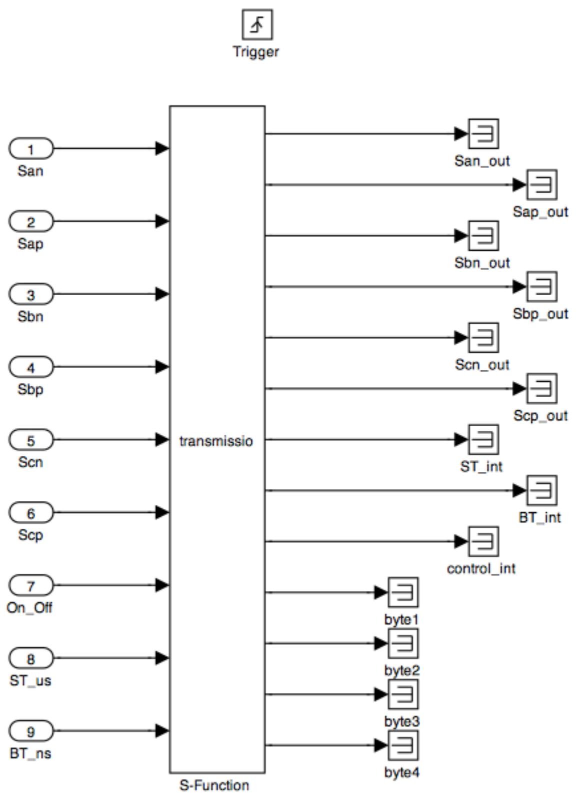


Fig. 11.5: Transmissio/Triggered Block inside

Now C code of *transmissio.c* is reported.

```
#define S_FUNCTION_NAME transmissio
#define S_FUNCTION_LEVEL 2

#include "simstruc.h"
#include <math.h>
```

```
#include <assert.h>
#include <stdlib.h>

#ifndef MATLAB_MEX_FILE
# include <ds1103.h>
# include <io1103.h>
#endif

#define pi 3.14159265
#define freq_fpga 25.125

typedef struct {                          /* structure sefinition ssed for sending */
  char_T byte1, byte2, byte3, byte4;     /* informations to the FPGA          */
} ByteStruct;

typedef union {
  UINT32_T word_out;
  ByteStruct quatre_bytes;
} sortida_master;

/* 'sortida_master' is an union formed by word_out (uInt32) and quatre_bytes.byte1,...
byte4 (char_T = byte)
The union let that word_out and quatre_bytes share same memory space. So, writing one
overwrite the other */


/* I/O DEFINITION */

static void mdlInitializeSizes(SimStruct *S)
{
  ssSetNumSFcnParams(S, 0);
  if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount (S)) {
    return;
  }

/* 9 Inputs, (Engegada, Sxy(6 bits = 6 inputs), ST_time_us, BT_time_ns) */
  if (!ssSetNumInputPorts(S, 9)) return;
  { int_T i;
    for (i=0; i<9; i++) {
      ssSetInputPortWidth(S, i, 1);
      ssSetInputPortDirectFeedThrough (S, i, 1);
    }
  }

/* 9 Outputs
(Sxy(6 bits = 6 outputs), ST_int(8 bits as int), BT_int(8 bits as int), Control_int(8 bits as
int), byte1, byte2, byte3, byte4) */
  if (!ssSetNumOutputPorts(S, 13)) return;
  { int_T j;
    for (j=0; j<13; j++) {
      ssSetOutputPortWidth(S,j,1);
    }
  }
  ssSetNumSampleTimes(S, 1);
  ssSetOptions(S, SS_OPTION_EXCEPTION_FREE_CODE |
SS_OPTION_USE_TLC_WITH_ACCELERATOR |SS_OPTION_PLACE_ASAP);
```

```
}

/*SAMPLE TIME INIT*/

static void mdlInitializeSampleTimes(SimStruct *S)
{
 ssSetSampleTime (S, 0, INHERITED_SAMPLE_TIME);
 ssSetOffsetTime (S, 0, 0.0);
}

/*I/O DIGITAL INIT OF DS1103 MASTER DSP */

#define MDL_START
#if defined (MDL_START)
static void mdlStart (SimStruct *S)
{
#ifndef MATLAB_MEX_FILE
 /* Configuration as Input/Output of 8 bits groups that form 32 bits master bus
 group 01 correspond to byte 04*/
 ds1103_bit_io_init (DS1103_DIO1_IN | DS1103_DIO2_OUT | DS1103_DIO3_OUT |
DS1103_DIO4_OUT);
#endif
}
#endif

/* MAIN FUNCTION */

static void mdlOutputs (SimStruct *S, int_T tid)
{


 /* VARIABLES */
 sortida_master  bits_32; /* 32 bits word that go to FPGA */

 real_T SamplingTime_us, BlankingTime_ns, ST_real;
 int_T j, Control_int, ST_int, BT_int;
 real_T On_Off;
 real_T San, Sap, Sbn, Sbp, Scn, Scp;


 InputRealPtrsType uPtrs0 = ssGetInputPortRealSignalPtrs(S, 0);
 InputRealPtrsType uPtrs1 = ssGetInputPortRealSignalPtrs(S, 1);
 InputRealPtrsType uPtrs2 = ssGetInputPortRealSignalPtrs(S, 2);
 InputRealPtrsType uPtrs3 = ssGetInputPortRealSignalPtrs(S, 3);
 InputRealPtrsType uPtrs4 = ssGetInputPortRealSignalPtrs(S, 4);
 InputRealPtrsType uPtrs5 = ssGetInputPortRealSignalPtrs(S, 5);
 InputRealPtrsType uPtrs6 = ssGetInputPortRealSignalPtrs(S, 6);
 InputRealPtrsType uPtrs7 = ssGetInputPortRealSignalPtrs(S, 7);
 InputRealPtrsType uPtrs8 = ssGetInputPortRealSignalPtrs(S, 8);

 real_T *y0 = ssGetOutputPortRealSignal(S,0);
 real_T *y1 = ssGetOutputPortRealSignal(S,1);
 real_T *y2 = ssGetOutputPortRealSignal(S,2);
 real_T *y3 = ssGetOutputPortRealSignal(S,3);
 real_T *y4 = ssGetOutputPortRealSignal(S,4);
 real_T *y5 = ssGetOutputPortRealSignal(S,5);
 real_T *y6 = ssGetOutputPortRealSignal(S,6);
 real_T *y7 = ssGetOutputPortRealSignal(S,7);
 real_T *y8 = ssGetOutputPortRealSignal(S,8);
 real_T *y9 = ssGetOutputPortRealSignal(S,9);
 real_T *y10 = ssGetOutputPortRealSignal(S,10);
 real_T *y11 = ssGetOutputPortRealSignal(S,11);
 real_T *y12 = ssGetOutputPortRealSignal(S,12);


 /* INPUTS */
 San = *uPtrs0[0];
 Sap = *uPtrs1[0];
 Sbn = *uPtrs2[0];
 Sbp = *uPtrs3[0];
 Scn = *uPtrs4[0];
 Scp = *uPtrs5[0];
 On_Off = *uPtrs6[0];
 SamplingTime_us = *uPtrs7[0];
 BlankingTime_ns = *uPtrs8[0];




 if (On_Off == 1.0) {

  Control_int = (int_T) (64+16);
  bits_32.quatre_bytes.byte1 = (char_T) Control_int;
  bits_32.quatre_bytes.byte2 = (char_T) (32*Scp+16*Scn+8*Sbp+4*Sbn+2*Sap+San);
  bits_32.quatre_bytes.byte3 = (char_T) 0;
  //bits_32.quatre_bytes.byte4 = (char_T) 0;

#ifndef MATLAB_MEX_FILE
  ds1103_bit_io_write(bits_32.word_out);     /* Synchronous Transmission of 32 bits word
                                               .word_out --> uInt32 type*/
#endif

  for (j=0;j<8000;j++){                  /* Wait 10 us  */
  }

  Control_int = (int_T) (64); // On_Off = 1, Enviament = 0.
  bits_32.quatre_bytes.byte1 = (char_T) Control_int;

#ifndef MATLAB_MEX_FILE
  ds1103_bit_io_write(bits_32.word_out);
#endif

 }

 else {
  Control_int = (int_T) (16);
  ST_real = ((freq_fpga*SamplingTime_us/256.0)-1.0);
  ST_int = (int_T) floor((freq_fpga*SamplingTime_us/256.0)-1.0 + 0.5); // ST_int =
freq_fpga*SamplingTime_us/256 - 1
  BT_int = (int_T)((BlankingTime_ns*freq_fpga/1000.0));
  bits_32.quatre_bytes.byte1 = (char_T) Control_int;
  bits_32.quatre_bytes.byte2 = (char_T) BT_int;
```

```
   bits_32.quatre_bytes.byte3 = (char_T) ST_int;
   //bits_32.quatre_bytes.byte4 = (char_T) 0;

#ifndef MATLAB_MEX_FILE
   ds1103_bit_io_write(bits_32.word_out);
#endif

   for (j=0;j<8000;j++){                    /* Espera 10 us  */
   }

   Control_int = (int_T) (0); // On_Off = 0, Enviament = 0.
   bits_32.quatre_bytes.byte1 = (char_T) Control_int;

#ifndef MATLAB_MEX_FILE
   ds1103_bit_io_write(bits_32.word_out);     /* Enviem l'ordre per abaixar la senyal
d'enviament */
#endif

  }

 y0[0] = San;
 y1[0] = Sap;
 y2[0] = Sbn;
 y3[0] = Sbp;
 y4[0] = Scn;
 y5[0] = Scp;
 y6[0] = ST_int;
 y7[0] = BT_int;
 y8[0] = Control_int;
 y9[0] = (uchar_T) bits_32.quatre_bytes.byte1;
 y10[0] = (uchar_T) bits_32.quatre_bytes.byte2;
 y11[0] = (uchar_T) bits_32.quatre_bytes.byte3;
 y12[0] = (uchar_T) bits_32.quatre_bytes.byte4;
}


static void mdlTerminate (SimStruct *S)           /* Ending */
{
}

#ifdef MATLAB_MEX_FILE
#include "simulink.c"
#else
#include "cg_sfun.h"
#endif
```

11.5. Predictive Control

This block is the heart of the control. Receive as input values of reference and measured currents, and measured values of 3-phase load voltages and DC link capacitors voltages.

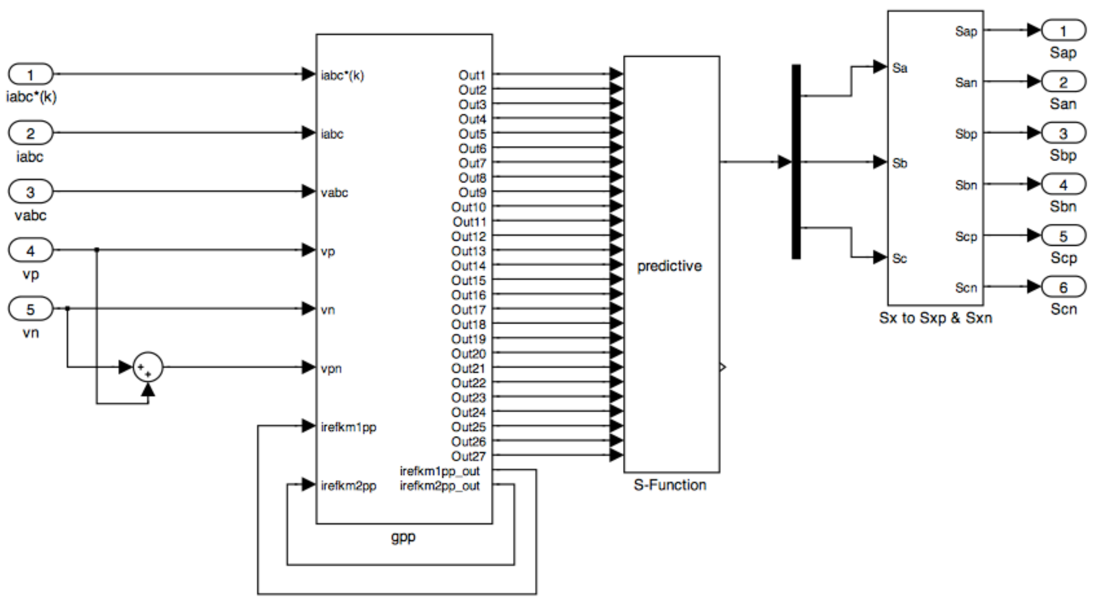


Fig. 11.6: Predictive Control Block

The main function of the block, is to read all the input data, calculate the optimum quality function for everyone of the 27 switching state, evaluate the best one and send the corresponding output configuration to the *trasmissio/triggered* block, ready for sending to the drivers of the IGBT.

Now the system is described looking separately every block.

11.5.1. *gpp* block

Next figure show the inside scheme of *gpp* block.

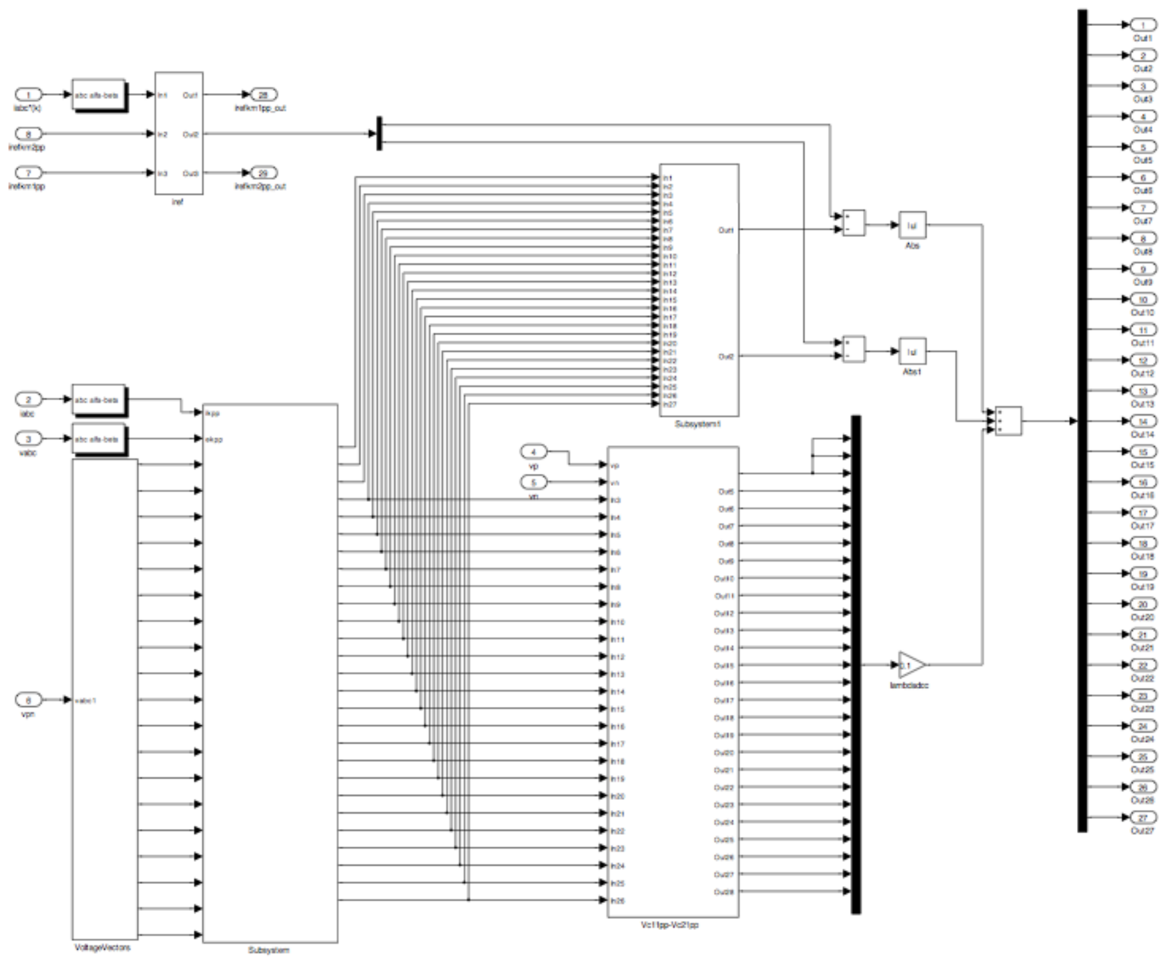


Fig. 11.7: gpp block

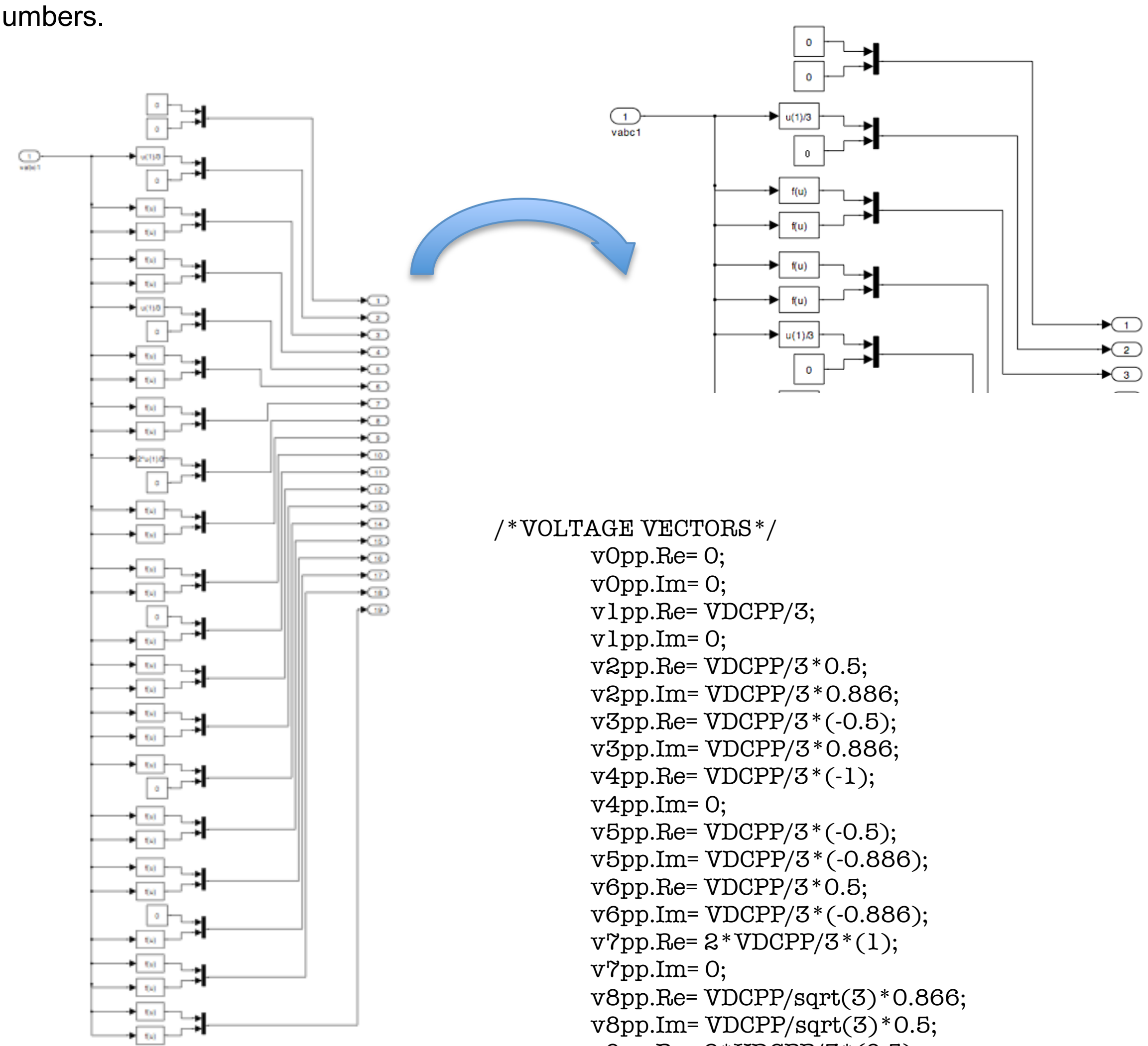
This really big image shows how many different blocks there is inside the gpp subsystem. The goal is to solve next equation with the result of calculate the best quality function for every voltage vector:

$$g_i = \left|i_\alpha^*(k+1) - i_\alpha(k+1)\right| + \left|i_\beta^*(k+1) - i_\beta(k+1)\right| + \lambda_{dc} \cdot \left|V_{c1}(k+1) - V_{c2}(k+1)\right|$$

with $i \in \{1,...,27\}$.

But go ahead with order can help to understand well every pass.

First of all, *Voltage Vector* block creates all the 19 possible voltage vectors generated by a three-level inverter. This means take the value of $V_{pn}=V_p-V_n$ like input, and multiply it for corresponding values, obtaining 19 complex numbers.



where VDCPP= vabc1.

```
/*VOLTAGE VECTORS*/
        v0pp.Re= 0;
        v0pp.Im= 0;
        v1pp.Re= VDCPP/3;
        v1pp.Im= 0;
        v2pp.Re= VDCPP/3*0.5;
        v2pp.Im= VDCPP/3*0.886;
        v3pp.Re= VDCPP/3*(-0.5);
        v3pp.Im= VDCPP/3*0.886;
        v4pp.Re= VDCPP/3*(-1);
        v4pp.Im= 0;
        v5pp.Re= VDCPP/3*(-0.5);
        v5pp.Im= VDCPP/3*(-0.886);
        v6pp.Re= VDCPP/3*0.5;
        v6pp.Im= VDCPP/3*(-0.886);
        v7pp.Re= 2*VDCPP/3*(1);
        v7pp.Im= 0;
        v8pp.Re= VDCPP/sqrt(3)*0.866;
        v8pp.Im= VDCPP/sqrt(3)*0.5;
        v9pp.Re= 2*VDCPP/3*(0.5);
        v9pp.Im= 2*VDCPP/3*0.886;
        v10pp.Re= 0;
        v10pp.Im= VDCPP/sqrt(3);
        v11pp.Re= 2*VDCPP/3*(-0.5);
        v11pp.Im= 2*VDCPP/3*0.886;
        v12pp.Re= VDCPP/sqrt(3)*(-0.866);
        v12pp.Im= VDCPP/sqrt(3)*0.5;
        v13pp.Re= 2*VDCPP/3*(-1);
        v13pp.Im= 0;
        v14pp.Re= VDCPP/sqrt(3)*(-0.866);
```

Once that these 19 voltage vectors are created, next goal is to calculate current prediction for every switching state, function realized from *Subsystem* block, that has like inputs the 19 voltage vector and alfa-beta transformation of measure currents and voltages. These reading values are passed like sampling values *ekpp* and *ikpp*, complex values for voltages and currents measured directly in the converter's IGBTs.

Equation realized is:

$$ik1_{pp,i} = \frac{L_{pp}}{R_{pp} \cdot T_{spp} + L_{pp}} \cdot ik_{pp} - \frac{T_{spp}}{R_{pp} \cdot T_{spp} + L_{pp}} \cdot ek_{pp} + \frac{T_{spp}}{R_{pp} \cdot T_{spp} + L_{pp}} \cdot v_{pp}(i)$$

where L_{pp} and R_{pp} are the values of the inductance and resistor of the LR filter, T_{spp} is the sampling period and vpp(i) is the voltage vector corresponding to switching state i.

Block *Vc11pp-Vc22pp* realizes next equation, fundamental for the quality function.

$$abs(Vc11pp - Vc21pp)$$

with

$$Vc11pp = V_p + \left(\frac{1}{C_{1pp}}\right) \cdot i_{C1pp} \cdot T_{spp}$$

$$Vc21pp = V_n + \left(\frac{1}{C_{2pp}}\right) \cdot i_{C2pp} \cdot T_{spp}$$

V_p and V_n are measured values of respectively upper and lower capacitor voltages of DC-link, C_{1pp}= C_{2pp}=1100μF is the value of both capacitors, T_{spp} is the sampling time and i_{c1pp} and i_{c2pp} are capacitors current related to every switching state prediction current.

We made the prediction for every phase current starting from $ik1_{pp,i}$ evaluated in previous block *Subsystem*, and calculating next system:

$$i_{app,i} = \mathrm{Re}\left\{ik1_{pp,i}\right\};$$

$$i_{bpp,i} = -0.5 \cdot \mathrm{Re}\left\{ik1_{pp,i}\right\} + \frac{\sqrt{3}}{2} \cdot \mathrm{Im}\left\{ik1_{pp,i}\right\};$$

$$i_{bpp,i} = -0.5 \cdot \mathrm{Re}\left\{ik1_{pp,i}\right\} - \frac{\sqrt{3}}{2} \cdot \mathrm{Im}\left\{ik1_{pp,i}\right\};$$

Now, for every possible switching state, we calculate the DC-link currents, depending on which voltage vector is related to that switching state. Next table show this relation and DC-link currents calculation.

Switching State	***Voltage Vector***	***i0pp***	***ippp***	***inpp***
[-1 -1 -1]	V0	0	0	0
[0 0 0]	V0	0	0	0
[1 1 1]	V0	0	0	0
[1 0 0]	V1	icpp+ibpp	iapp	0
[0 -1 -1]	V1	iapp	0	ibpp+icpp
[1 1 0]	V2	icpp	iapp+ibpp	0
[0 0 -1]	V2	iapp+ibpp	0	icpp
[0 1 0]	V3	iapp+icpp	ibpp	0
[-1 0 -1]	V3	ibpp	0	iapp+icpp
[0 1 1]	V4	iapp	ibpp+icpp	0
[-1 0 0]	V4	ibpp+icpp	0	iapp
[0 0 1]	V5	iapp+ibpp	icpp	0
[-1 -1 0]	V5	icpp	0	iapp+ibpp
[1 0 1]	V6	ibpp	iapp+icpp	0
[0 -1 0]	V6	iapp+icpp	0	ibpp
[1 -1 -1]	V7	0	iapp	ibpp+icpp
[1 0 -1]	V8	ibpp	iapp	icpp
[1 1 -1]	V9	0	iapp+ibpp	icpp
[0 1 -1]	V10	iapp	ibpp	icpp
[-1 1 -1]	V11	0	ibpp	iapp+icpp
[-1 1 0]	V12	icpp	ibpp	iapp
[-1 1 1]	V13	0	ibpp+icpp	iapp
[-1 0 1]	V14	ibpp	icpp	iapp
[-1 -1 1]	V15	0	icpp	iapp+ibpp
[0 -1 1]	V16	iapp	icpp	ibpp
[1 -1 1]	V17	0	iapp+icpp	ibpp
[1 -1 0]	V18	icpp	iapp	ibpp

Tab. 11.1: Swiching state DC-link currents

Finally, i_{c1pp}= -ippp and i_{c2pp}= inpp.

Now only one block lacks, the reference extrapolation block *iref*.

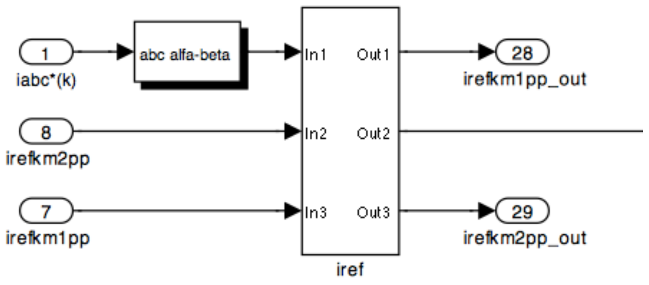


Fig. 11.8: iref block

This block creates the reference current *irefk1pp* needed in the calculation on the quality function *g.*

irefk1pp is a second order extrapolation on the value of the reference current generated in *current reference abc* block, using the equation:

$$irefk1pp = 3 \cdot irefkpp - 3 \cdot irefkm1pp + irefkm2pp$$

where *irefkpp* is the alfa-beta transformation of the the reference current in the sampling istant (k), *irefkm1pp* is the reference current in instant (*k-1*) and *irefkm2pp* in instant (*k*-*2*). Obviously, *irefk1pp* is referred to future instant (*k*+*1*).

After every calculation step of *irefk1pp*, the values of *irefkm1pp and irefkm2pp* are reassigned in this way.

$$irefkm2pp = irefkm1pp$$
$$irefkm1pp = irefkpp$$

Next figure shows how this has been realized.

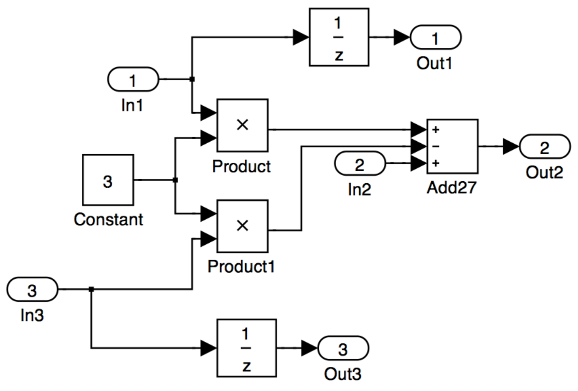


Fig. 11.9: Reference Current Extrapolation

We are now ready for evaluating the 27 quality function, corresponding to equation (a.1), and this is realized from blocks Subsytem1, Vc11pp-Vc22pp and final stage of add, abs and multiplexers. The result are the 27 quality functions evaluate for every switching state.

11.5.2. S-Function Predictive Block

This block only takes all 27 calculated quality functions, and evaluate which one is the smallest, so the best quality function, and pass to next block the corresponding switching state.

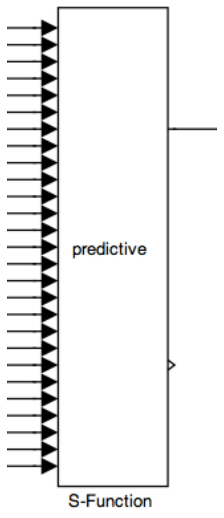


Fig. 11.10: predictive block

C code is now shown.

```
#define S_FUNCTION_NAME predictive
#define S_FUNCTION_LEVEL 2

#include "simstruc.h"
#include <math.h>
#include <assert.h>
#include <stdlib.h>


/* I/O DEFINITION */

static void mdlInitializeSizes(SimStruct *S)
{
	ssSetNumSFcnParams(S, 0);
	if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount (S)) {
   return;
  }

	/*5+8 INPUTS (irefkpp, ikpp, Vc1cpp, Vc2cpp, ekpp, constantes, irefkm1pp,
irefkm2pp) */
	if (!ssSetNumInputPorts(S,27))
		return;

	{
      int_T i;
		for (i=0; i<=26; i++)
        {
          ssSetInputPortWidth(S, i, 1);
          ssSetInputPortDirectFeedThrough (S, i, 1);
        }

	}



	/* 3+1 OUTPUTS (xpp, irefkm1pp, irefkm2pp, prueba)*/

	if (!ssSetNumOutputPorts(S, 2))
		return;

	{
		ssSetOutputPortWidth(S,0,3);
      /*pruebas*/
      ssSetOutputPortWidth(S,1,1);
	}

	ssSetNumSampleTimes(S, 1);
	ssSetOptions(S, SS_OPTION_EXCEPTION_FREE_CODE |
		SS_OPTION_USE_TLC_WITH_ACCELERATOR |SS_OPTION_PLACE_ASAP);
}

/* SAMPLING TIME INIT*/
```

```
        static void mdlInitializeSampleTimes(SimStruct *S)
        {
                ssSetSampleTime (S, 0, INHERITED_SAMPLE_TIME);
                ssSetOffsetTime (S, 0, 0.0);
        }

/*MAIN FUNCTION*/
static void mdlOutputs (SimStruct *S, int_T tid)
{
  int_T  xoppp, j;
  /*real_T goppp, gpp1, gpp2, gpp3, gpp4, gpp5, gpp6, gpp7, gpp8, gpp9, gpp10;
  real_T gpp11, gpp12, gpp13, gpp14, gpp15, gpp16, gpp17, gpp18, gpp19, gpp20;
  real_T gpp21, gpp22, gpp23, gpp24, gpp25, gpp26, gpp27; */
  real_T gpp[27];
  real_T goppp;

        int_T  estadospp[27][3]=
        {
                {0, 0, 0},{-1, -1, -1},{1, 1, 1},{1, 0, 0},{0, -1, -1},{1, 1, 0},{0, 0, -1},
                {0, 1, 0},{-1, 0, -1},{0, 1, 1},{-1, 0, 0},{0, 0, 1},{-1, -1, 0},{1, 0, 1},
                {0, -1, 0},{1, -1, -1},{1, 0, -1},{1, 1, -1},{0, 1, -1},{-1, 1, -1},{-1, 1, 0},
                {-1, 1, 1},{-1, 0, 1},{-1, -1, 1},{0, -1, 1},{1, -1, 1},{1, -1, 0}
        };



        /* Data adquisition */
        InputRealPtrsType uPtrs0 = ssGetInputPortRealSignalPtrs(S, 0);
        InputRealPtrsType uPtrs1 = ssGetInputPortRealSignalPtrs(S, 1);
        InputRealPtrsType uPtrs2 = ssGetInputPortRealSignalPtrs(S, 2);
        InputRealPtrsType uPtrs3 = ssGetInputPortRealSignalPtrs(S, 3);
        InputRealPtrsType uPtrs4 = ssGetInputPortRealSignalPtrs(S, 4);
        InputRealPtrsType uPtrs5 = ssGetInputPortRealSignalPtrs(S, 5);
        InputRealPtrsType uPtrs6 = ssGetInputPortRealSignalPtrs(S, 6);
        InputRealPtrsType uPtrs7 = ssGetInputPortRealSignalPtrs(S, 7);
        InputRealPtrsType uPtrs8 = ssGetInputPortRealSignalPtrs(S, 8);
        InputRealPtrsType uPtrs9 = ssGetInputPortRealSignalPtrs(S, 9);
        InputRealPtrsType uPtrs10 = ssGetInputPortRealSignalPtrs(S, 10);
        InputRealPtrsType uPtrs11 = ssGetInputPortRealSignalPtrs(S, 11);
        InputRealPtrsType uPtrs12 = ssGetInputPortRealSignalPtrs(S, 12);
        InputRealPtrsType uPtrs13 = ssGetInputPortRealSignalPtrs(S, 13);
        InputRealPtrsType uPtrs14 = ssGetInputPortRealSignalPtrs(S, 14);
        InputRealPtrsType uPtrs15 = ssGetInputPortRealSignalPtrs(S, 15);
        InputRealPtrsType uPtrs16 = ssGetInputPortRealSignalPtrs(S, 16);
        InputRealPtrsType uPtrs17 = ssGetInputPortRealSignalPtrs(S, 17);
        InputRealPtrsType uPtrs18 = ssGetInputPortRealSignalPtrs(S, 18);
        InputRealPtrsType uPtrs19 = ssGetInputPortRealSignalPtrs(S, 19);
        InputRealPtrsType uPtrs20 = ssGetInputPortRealSignalPtrs(S, 20);
        InputRealPtrsType uPtrs21 = ssGetInputPortRealSignalPtrs(S, 21);
        InputRealPtrsType uPtrs22 = ssGetInputPortRealSignalPtrs(S, 22);
        InputRealPtrsType uPtrs23 = ssGetInputPortRealSignalPtrs(S, 23);
        InputRealPtrsType uPtrs24 = ssGetInputPortRealSignalPtrs(S, 24);
        InputRealPtrsType uPtrs25 = ssGetInputPortRealSignalPtrs(S, 25);
        InputRealPtrsType uPtrs26 = ssGetInputPortRealSignalPtrs(S, 26);
```

```
            /* output selection */
            real_T *y0 = ssGetOutputPortRealSignal(S,0);
            real_T *y1 = ssGetOutputPortRealSignal(S,1);

   gpp[0]=  *uPtrs0[0];
   gpp[1]=  *uPtrs1[0];
   gpp[2]=  *uPtrs2[0];
   gpp[3]=  *uPtrs3[0];
   gpp[4]=  *uPtrs4[0];
   gpp[5]=  *uPtrs5[0];
   gpp[6]=  *uPtrs6[0];
   gpp[7]=  *uPtrs7[0];
   gpp[8]=  *uPtrs8[0];
   gpp[9]=  *uPtrs9[0];
   gpp[10]=  *uPtrs10[0];
   gpp[11]=  *uPtrs11[0];
   gpp[12]=  *uPtrs12[0];
   gpp[13]=  *uPtrs13[0];
   gpp[14]=  *uPtrs14[0];
   gpp[15]=  *uPtrs15[0];
   gpp[16]=  *uPtrs16[0];
   gpp[17]=  *uPtrs17[0];
   gpp[18]=  *uPtrs18[0];
   gpp[19]=  *uPtrs19[0];
   gpp[20]=  *uPtrs20[0];
   gpp[21]=  *uPtrs21[0];
   gpp[22]=  *uPtrs22[0];
   gpp[23]=  *uPtrs23[0];
   gpp[24]=  *uPtrs24[0];
   gpp[25]=  *uPtrs25[0];
   gpp[26]=  *uPtrs26[0];


   goppp=1000;

           for (j=0; j<=26; j++)
       {
         if (gpp[j]<goppp)
         {
            xoppp=j;
            goppp=gpp[j];
         }

       }
   y0[0]=estadospp[xoppp][0];
   y0[1]=estadospp[xoppp][1];
   y0[2]=estadospp[xoppp][2];
   y1[0]=goppp;

}

static void mdlTerminate (SimStruct *S)
{
}

#ifdef MATLAB_MEX_FILE
```

```
#include "simulink.c"
#else
#include "cg_sfun.h"

#endif
```

All input quality functions are read, and the for cycle select the best one, saving a value only if is lower than the last.

Every input is related to a different switching state saved in the array *estadospp*, and in case of selection is passed to the output like a 3 dimension vector. Is also available a second output that pass the value of the best quality function, for an eventually analysis.

11.5.3. Sx to Sxp & Sxn Block

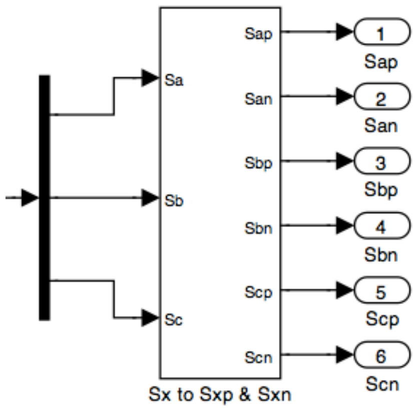


Fig. 11.11: Sx to Sxp & Sxn block

This block, thanks to 1-to-3 demultiplexer, take the switching state codification, and transforms it to the 6 values that after the trasmissio/triggered block receives and elaborates. Next figure shows how this block has been realized.

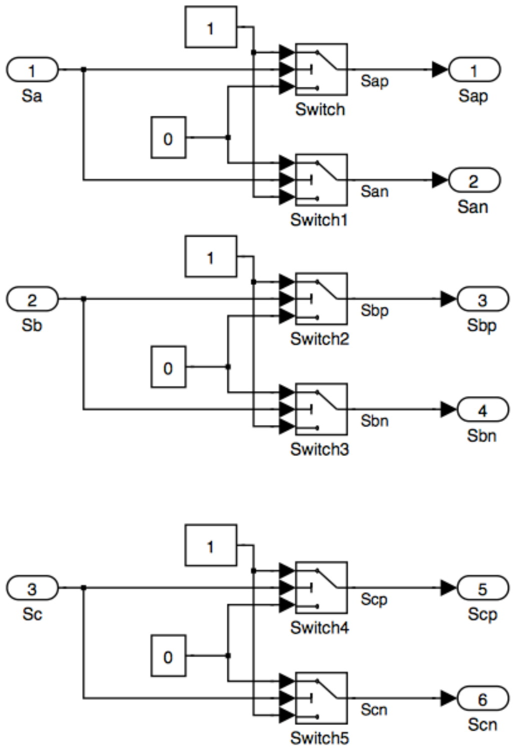


Fig. 11.12: Sx to Sxp & Sxn block inside

The functioning is really simple, and next true table shows it, where $i \in \{a, b, c\}$.

Si	**Sip**	**Sin**
1	1	0
0	0	0
-1	0	1

Tab. 11.2: True Table for block Sx to Sxp&Sxn

11.6. Control on/off + Errors

The function of this block is manage the commands of switch on and switch off of the converter, depending on the Control Desk user interface and on dSpace interruptions.

The state of the button On/Off present in the Control Desk, is multiply for Function-Call Delay block outputs. These output have an initial value of 1 and change to 0 (switching off the converter) if one Error Interrupt is activated. The Transmission Error is not used in this project.

Is necessary to reprogram the dSpace by the Control Desk interface if the converter has been switched off by interrupt activation.

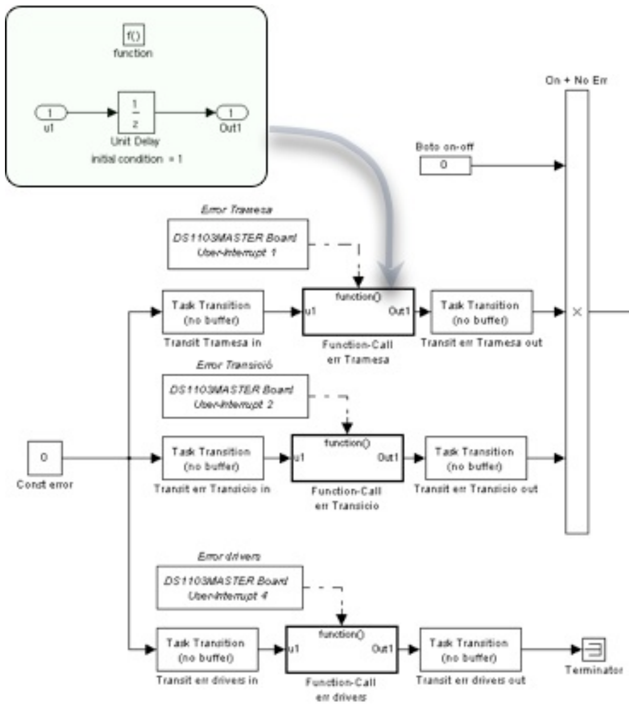


Fig. 11.13: Control in/off + errors block

11.7. PLL for network connection

No se puede mostrar la imagen. Puede que su equipo no tenga suficiente memoria para abrir la imagen o que ésta esté dañada. Reinicie el equipo y, a continuación, abra el archivo de nuevo. Si sigue apareciendo la x roja, puede que tenga que borrar la imagen e insertarla de nuevo.

Fig. 11.14: PLL for network synchronization

This block generate the *angle* in all experimental tests with network connection. Allow to a perfect synchronization between the converter and the three-phase generator that simulate the connection with the network.

12. FPGA Programming

The objective of this appendix is to explain how the control of the inverter has been realized by using the FPGA of UP2 Board from Altera. This is the responsible of final sending to IGBT drivers of control signals received from dSpace elaboration, using Digital Communication Board.

For the programming, Maxplus II Baseline v.10.2 has been used, with students free license.

Scope of the FPGA programming are:

- Insertion of the Blanking Time value rebut from the dSpace, useful to avoid short-circuit in the IGBTs during level transitions.
- Generation of a clock signal responsible of the updating of converter state when necessary.
- Decoding of switching states received from the dSpace.

12.1. Specifications

Specifications that FPGA programming must comply are:

- Possibility of changing the Blanking Time Value.
- Realize of all possible switching state in the correct order and using the Blanking Time value.
- Connect IGBTs drivers errors to dSpace interrupts.

12.2. Design

Design has been realized with Maxplus II software. Now all block will be described, starting from a complete view of the system shows in next Figure.

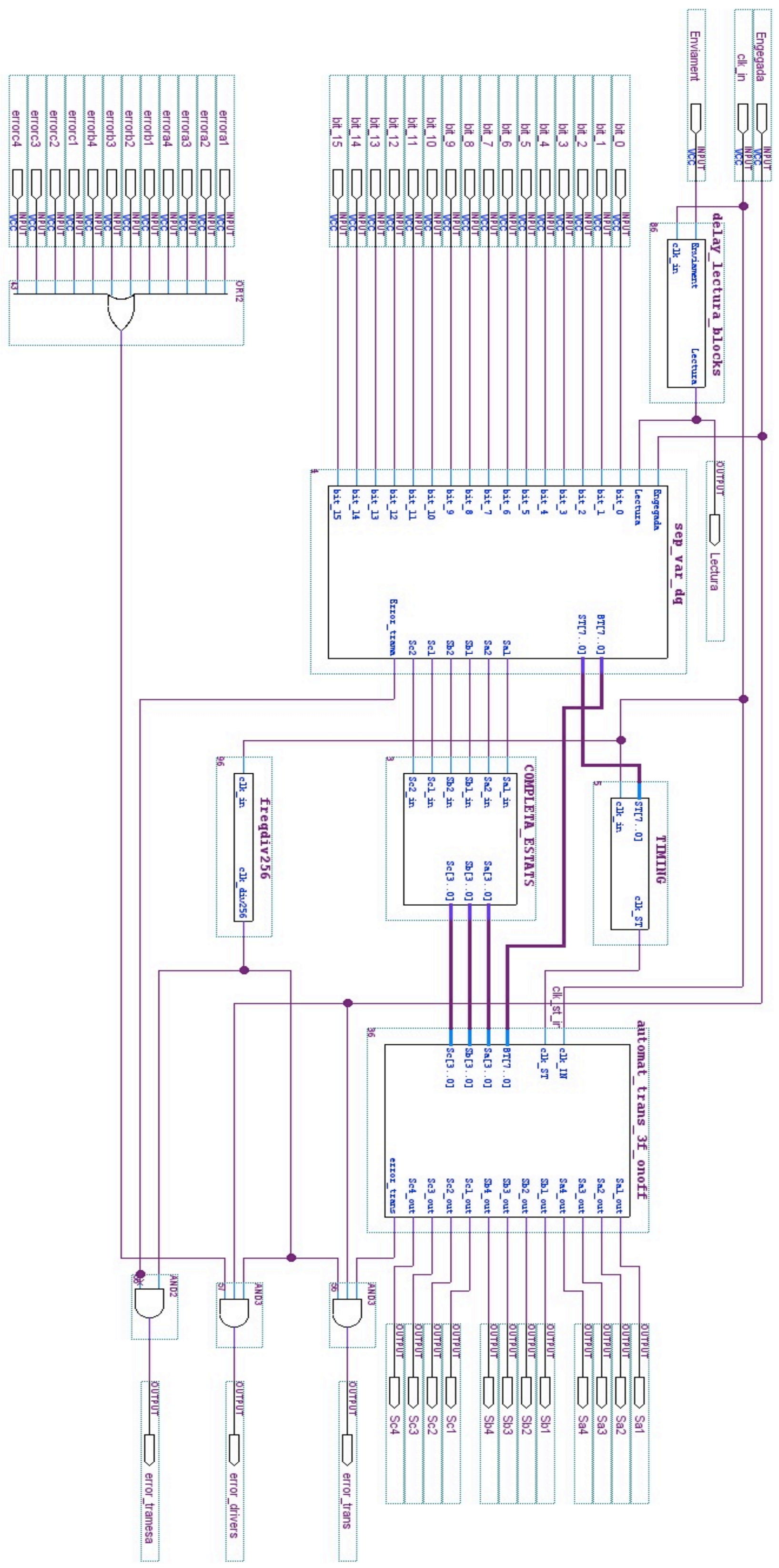


Fig. 12.1: entire FPGA diagram

12.2.1. Delay Lectura Block

This block add a delay between the activation of output *Lectura* and an *Enviament* rising egde. Waiting time is of 32 cycles of Clk_in, 25.125 MHz internal clock of FPGA (32/25.125 Mhz = 1.27 µs).

This delay create a reasonable time for signals stabilization of data bus before the reading process, and also help to “clean” the *Enviament* signal.

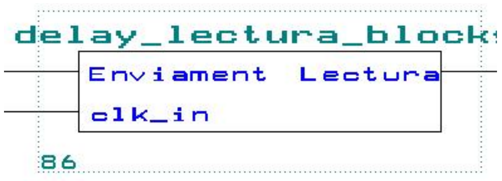


Fig. 12.2: Delay Lectura block

	INPUT
Enviament	Control signal. Indicates presence of new data in the digital I/O bus.
Clk_In	Internal Clock of the FPGA

Tab. 12.1: Input signals

	OUTPUT
Lectura	Control signal. Indicates that next block can read data.

Tab. 12.2: Output signals

This block is realized using various default blocks of Maxplus, like comparators and counters, and next figure shows the realization.

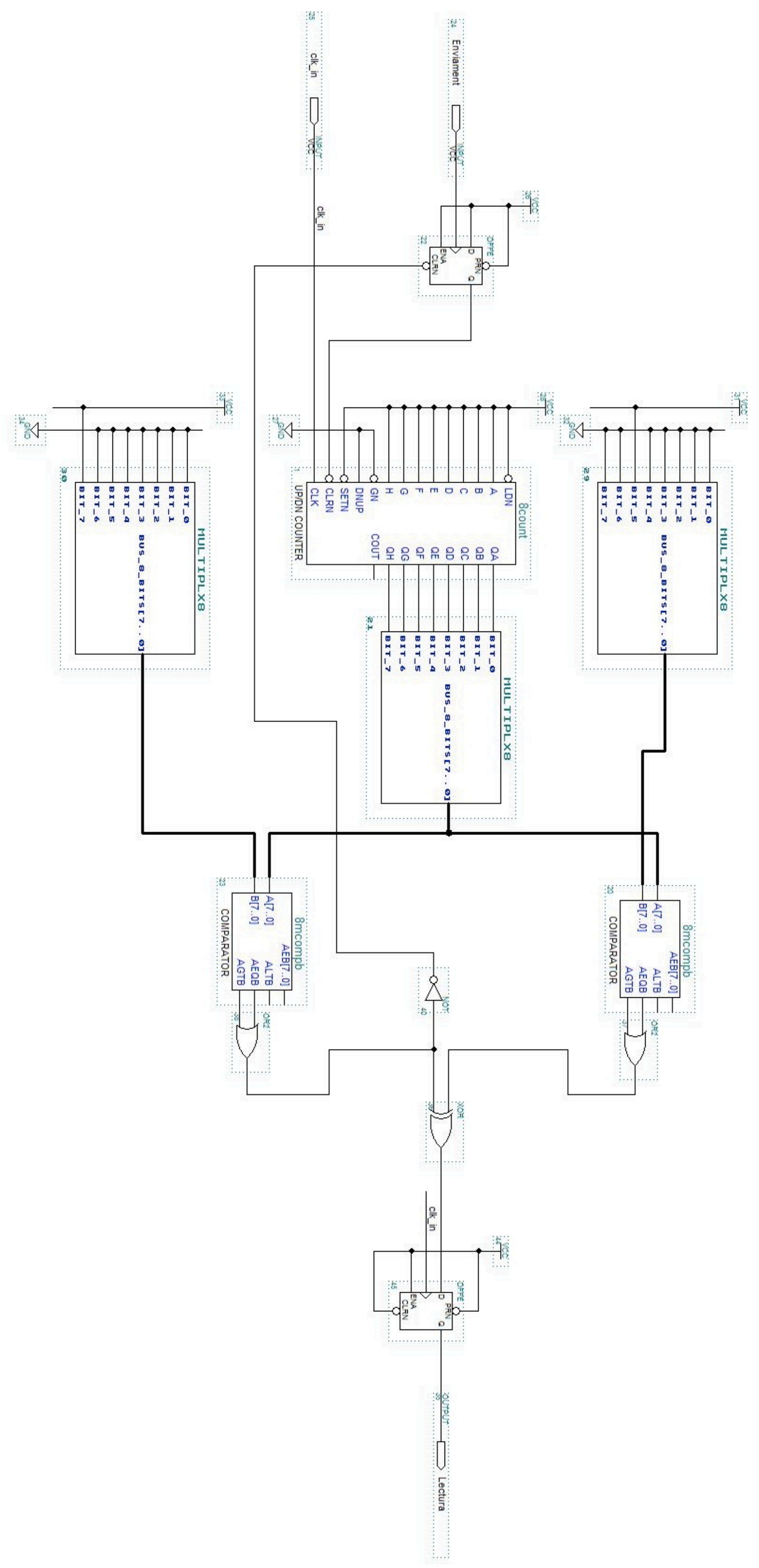


Fig. 12.3: Delay Lectura Block inside

12.2.2. Sep_var_dq Block

This block function is multiplex data received from the dSpace through outputs ST and BT. *Engegada* signal selects the output channel, and on rising edge of *Lectura* signal, data are copied to outputs.

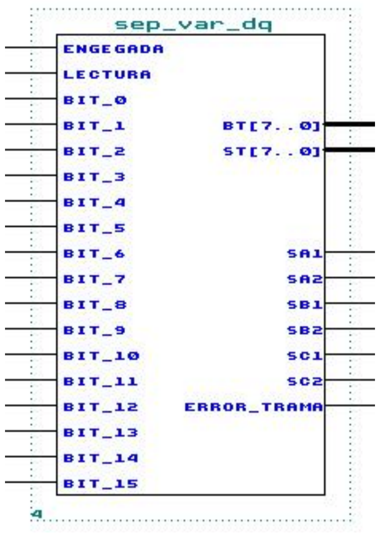


Fig. 12.4: Sep_var_dq block

	INPUT
Engegada	Sent from dSpace. State con converter signal. Multiplex bit_0…bit_15 data to Sxy if is on or to ST and BT if is off.
Lectura	Control Sinal. Activate copy of input data to output, depending on Engegada signal.
bit_0…bit_15	Bytes 2 and 3 of I/O digital bus.(32bits, 4 bytes)

Tab. 12.3: Input Signals

	OUTPUT
ST[7…0]	Frequency of Clk_in selector
BT[7…0]	Multiply factor of Clk_in cycles used for Blanking Time
Sa1,Sa2,Sb1, Sb2,Sc1,Sc2	Indicate connections between phases a,b,c with p,n levels. Connection to 0 level is decode in Completa_estat block
Error_trama	Alarm signal. Is activated when Sx1=1 and Sx2=1 are detected in one phase, on input bits 5…0

Tab. 12.4: Output signals

Next true table can resume the functioning of the block.

Eng.	Lec.	ST	BT	Sxy
0	x	old ST	old BT	1*
0	↑	bit 8…15	bit 0…7	1*
1	x	old ST	old BT	old Sxy
1	↑	old ST	old BT	bit 0…5

Tab. 12.5: True table for Sep_var_dq block. (* output =1 because Completa_estats can undestand that converter os switched off)

	ON	OFF
bit_0	San	BT[1]
bit_1	Sap	BT[2]
bit_2	Sbn	BT[3]
bit_3	Sbp	BT[4]
bit_4	Scn	BT[5]
bit_5	Scp	BT[6]
bit_6	-	BT[7]
bit_7	-	BT[8]
bit_8	-	ST[1]
bit_9	-	ST[2]
bit_10	-	ST[3]
bit_11	-	ST[4]
bit_12	-	ST[5]
bit_13	-	ST[6]
bit_14	-	ST[7]
bit_15	-	ST[8]

Tab. 12.6: Data Bus content (bytes 2 and 3) when converter is ON and OFF

Error_trama is an output signal activated when configuration Sx1=1 and Sx2=1 is present in the same moment than the converter is working. This create an error signal sent to a dSpace interrupt, forcing the converter switching off. Next table resume *Error_trama*value depending on input signals values.

Eng.	Lec.	Sx1 Sx2	Error_trama
0	x	x	0
0	↑	x	0
1	x	x	old ET
1	↑	1 1	1
1	↑	not (1 1)	0

Tab. 12.7: Error_trama true table

Next figure shows realization of this block.

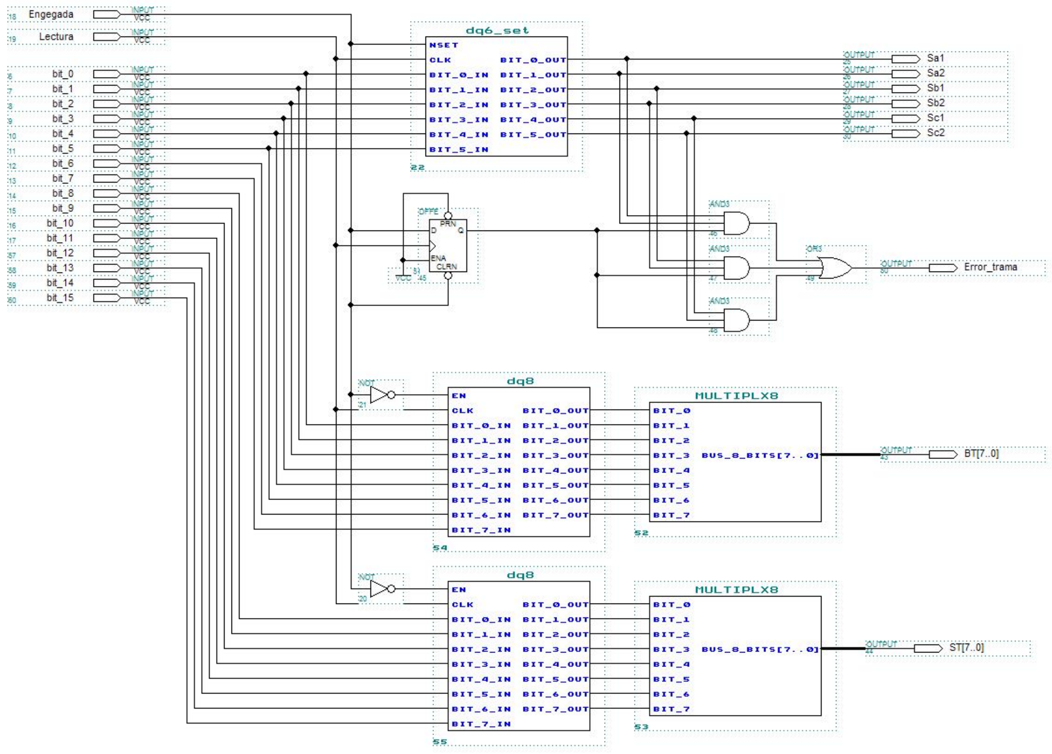


Fig. 12.5: Sep_var_dq block inside

12.2.3. Completa_Estats Block

This block function is decoding phase state bits arriving from dSpace (two every phase) into four signals that indicate the state of 4 transistor of every branch.

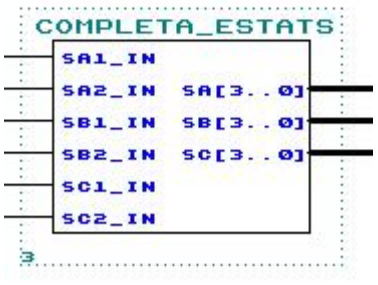


Fig. 12.6: Completa_estats block

	INPUT
Sa1_in, Sa2_in, Sb1_in, Sb2_in, Sc1_in, Sc2_in	indicate the value of every input to p or n level. these input describe to which level is connected every phase.

Tab. 12.8: Input Signals

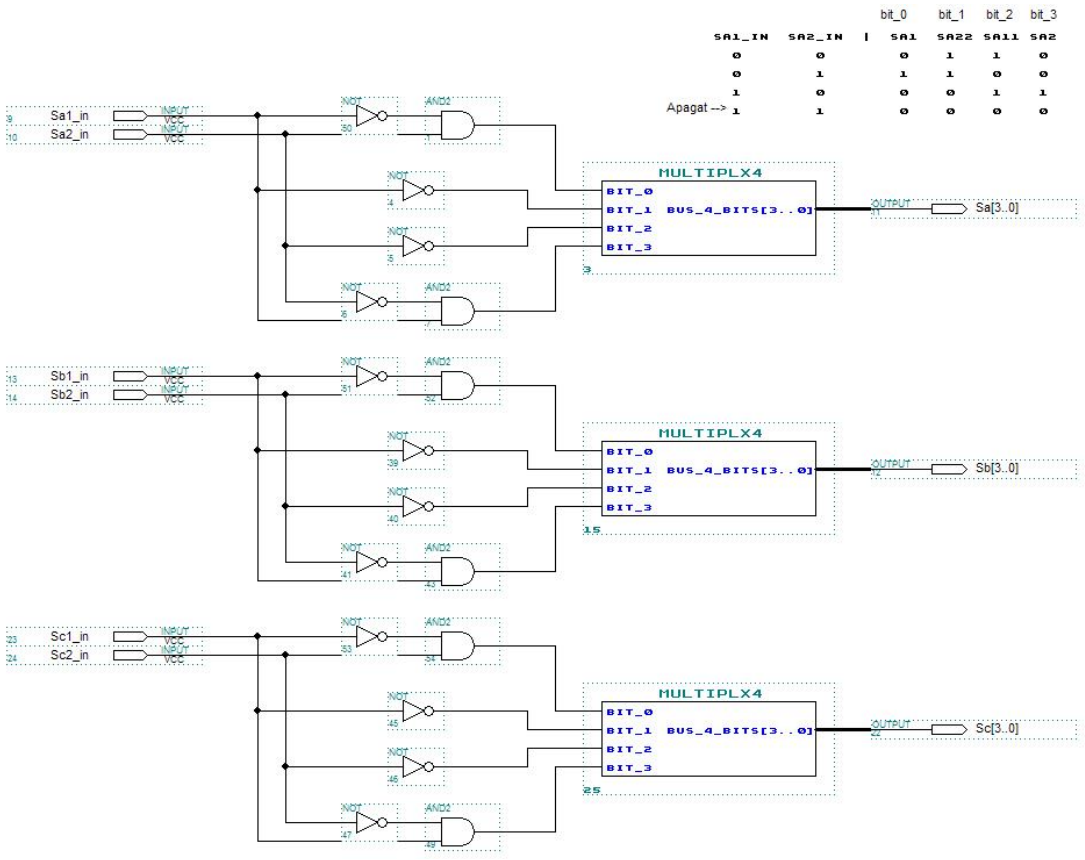
	OUTPUT
Sa[3...0], Sb[3...0], ca[3...0]	4 bits signals that describe the state of connection of every transistor of every branch.

Tab. 12.9: Output signals

Next table shows how the decoding is realized, and after fig. B.7. shows the internal design of the block.

Sx1_in	Sx2_in	Sx3	Sx2	Sx1	Sx0	Phase connection
0	0	0	1	1	0	O
0	1	1	1	0	0	p
1	0	0	0	1	1	n
1	1	0	0	0	0	Everything open

Tab. 12.10: Decoding table



			bit_0	bit_1	bit_2	bit_3
SA1_IN	SA2_IN	|	SA1	SA22	SA11	SA2
0	0		0	1	1	0
0	1		1	1	0	0
1	0		0	0	1	1
Apagat --> 1	1		0	0	0	0

Fig. 12.7: Completa_estats block inside

12.2.4. Automat_trans_3f_onoff block

This block is an automata that realized the transition between states, including the Blanking Time, for every IGBT.

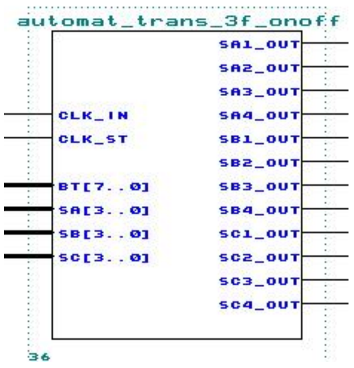


Fig. 12.8: Automat_trans Block

The automata, programmed in VHDL, is formed by 3 different processes that shared 2 signals. The elements of these 2 groups are:

1. Signals:

    - **e_actual**: converter current state ∈ {P, O, N, T1, T2, BTe, Apagat}

    - **e_objectiu**: goal state ∈ {P, O, N, Apagat}

2. Processes:

    - **Camí Transicions**: automata where, depending on signal e_objectiu, the value of e_actual is changed in every falling edge of Clk_IN. In this way correct transitions are realized, including the Blanking Time.

    - **Selecció Estat Objectiu**: process that actualize, in every falling edge of Clk_ST, the value of e_objectiu depending on inputs (Sa[3..0], Sb[3..0], Sc[3..0]).

Now the VHDL is shown.

```
LIBRARY ieee;
--
-- Import all the declarations in a package
USE ieee.std_logic_1164.all;
--USE ieee.std_logic_unsigned.ALL;
--USE ieee.std_logic_arith.all;
--
-- ENTITY
ENTITY automat_trans_1f_OnOff is
--
        PORT
        (
                Sx              : IN STD_LOGIC_VECTOR(3 downto 0);
                clk_IN          : IN STD_LOGIC;
                clk_ST          : IN STD_LOGIC;
                BT              : IN INTEGER RANGE 255 downto 0;
                S1_out, S2_out, S3_out, S4_out          : OUT STD_LOGIC
        );
END automat_trans_1f_OnOff;


ARCHITECTURE funcional of automat_trans_1f_OnOff is

TYPE    estats  IS (Apagat,P,O,N,T1,T2,BTs);
SIGNAL  e_objectiu      : estats := Apagat;
SIGNAL  e_actual        : estats := Apagat;
SIGNAL  e_obj_anterior : estats := Apagat;
```

```
BEGIN

sel_e_objectiu : PROCESS (clk_ST)
BEGIN

IF falling_edge(clk_ST) THEN

          e_obj_anterior <= e_objectiu;

          CASE Sx IS
          WHEN "1100" => e_objectiu <= N;
          WHEN "0110" => e_objectiu <= O;
          WHEN "0011" => e_objectiu <= P;
          WHEN "0000" => e_objectiu <= Apagat;
          WHEN OTHERS => e_objectiu <= e_obj_anterior;
          END CASE;
END IF;

END PROCESS sel_e_objectiu;

camins_transicions : PROCESS (clk_IN)
VARIABLE Sx_out                    : STD_LOGIC_VECTOR(3 downto 0);
VARIABLE PN_flag1                  : INTEGER;
VARIABLE PN_flag2                  : INTEGER;
VARIABLE bt_count                  : INTEGER RANGE 255 downto 0;
VARIABLE bt_count_state      : STD_LOGIC := '0';
BEGIN

IF (falling_edge(clk_IN)) THEN

                    IF e_objectiu = Apagat THEN

                    Sx_out := "0000";
                    CASE e_objectiu IS
                    WHEN O|P       => e_actual <= T1;
                    WHEN N                   => e_actual <= T2;
                    WHEN OTHERS              => e_actual <= Apagat;
                    END CASE;

          ELSE

                    CASE e_actual IS

                    WHEN Apagat =>
                              Sx_out := "0000";
                              CASE e_objectiu IS
                              WHEN O|P                 =>        e_actual <= T1;
                              WHEN N                   =>        e_actual <= T2;
                              WHEN OTHERS              =>        e_actual <= Apagat;
                              END CASE;

                    WHEN P =>
                              Sx_out := "1100";
                              CASE e_objectiu IS
                              WHEN O          =>        e_actual <= T1;
                              WHEN N          =>        PN_flag1 := 1;
```

```
                                PN_flag2 := 1;
                                e_actual <= T1;

             WHEN OTHERS =>     e_actual <= P;
             END CASE;

     WHEN T1 =>
             Sx_out := "0100";
             e_actual <= BTs;

     WHEN O =>
             Sx_out := "0110";
             IF (PN_flag2 = 1) THEN
                     e_actual <= BTs;
             ELSE
                     CASE e_objectiu IS
                     WHEN P         =>      e_actual <= T1;
                     WHEN N         =>      e_actual <= T2;
                     WHEN OTHERS =>         e_actual <= O;
                     END CASE;
             END IF;

     WHEN T2 =>
             Sx_out := "0010";
             e_actual <= BTs;

     WHEN N =>
             Sx_out := "0011";
             CASE e_objectiu IS
             WHEN P =>                  PN_flag1 := 1;
                                        PN_flag2 := 1;
                                        e_actual <= T2;

             WHEN O => e_actual <= T2;
             WHEN OTHERS => e_actual <= N;
             END CASE;

     WHEN BTs =>

             IF (bt_count < BT) THEN
                     bt_count := bt_count + 1;
             ELSE
                     bt_count := 0;

                     IF (PN_flag1 = 1) THEN
                             e_actual <= O;
                             PN_flag1 := 0;
                     ELSIF (PN_flag2 = 1) THEN

                             IF (e_objectiu = P) THEN e_actual <= T1;
                             ELSIF (e_objectiu = N) THEN e_actual <= T2;
                             END IF;
                     PN_flag2 := 0;

                     ELSIF (e_objectiu = P) THEN
                                     e_actual <= P;
```

```
                                        ELSIF (e_objectiu = O) THEN
                                                    e_actual <= O;

                                        ELSIF (e_objectiu = N) THEN
                                                    e_actual <= N;
                                        END IF;

                              END IF;

                    END CASE;

          END IF;

END IF;

S1_out <= Sx_out(3);  -- S1
S2_out <= Sx_out(2);  -- S22
S3_out <= Sx_out(1);  -- S11
S4_out <= Sx_out(0);  -- S2

END PROCESS camins_transicions;

END funcional;
```

To explain how it works some graphics can help.

12.3. *Camí Transicions* Process

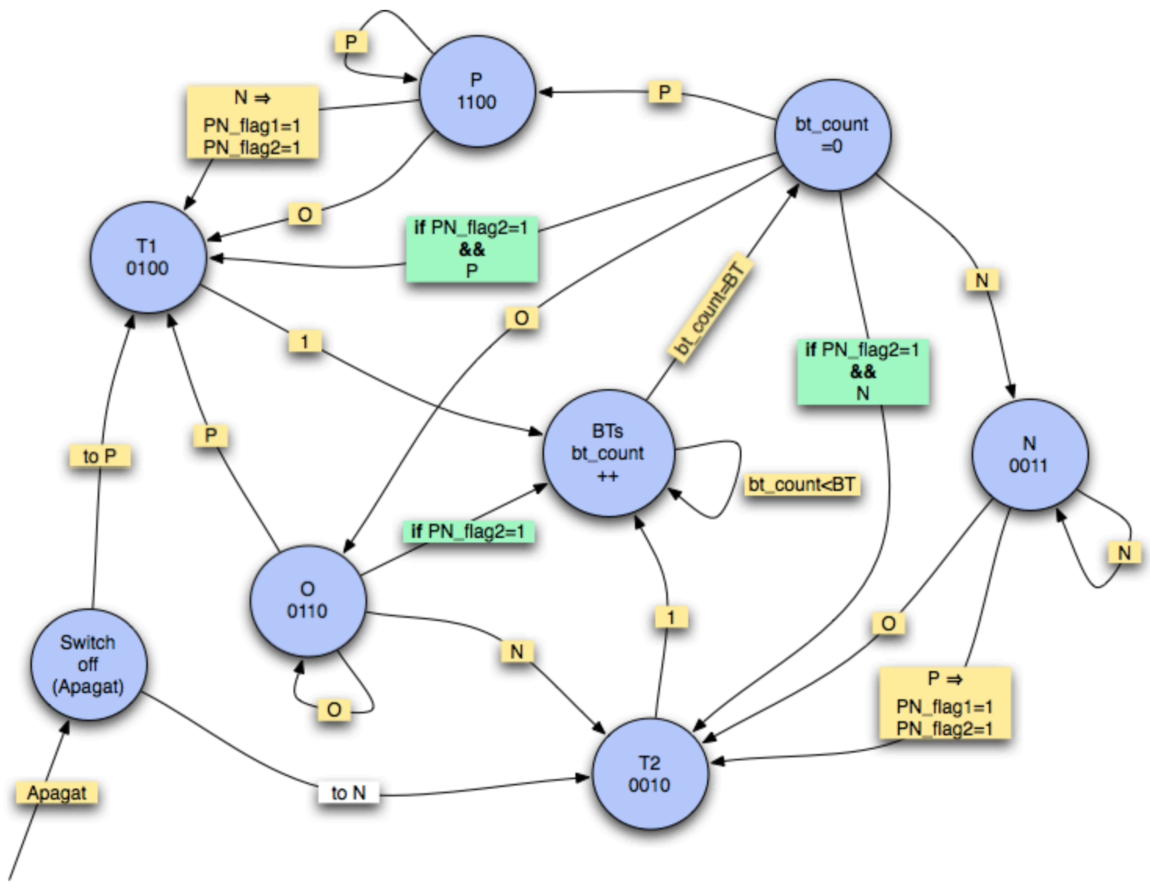


Fig. 12.9: *Camí Transicions* process

The circles represent states of the variable *e_actual*, and bits of *Sx_out*.

Transactions occur when the conditions indicated on the arrows are respected. The conditions may depend only by the variable *e_objectiu* (yellow background), the variable *e_objectiu* and the activation of the flags *PN_flag1* and *PN_flag2* (green background), or, in the case of the state BTs, the state of the variable *bt_count*.

12.4. *Selecció Estat Objectiu* Process

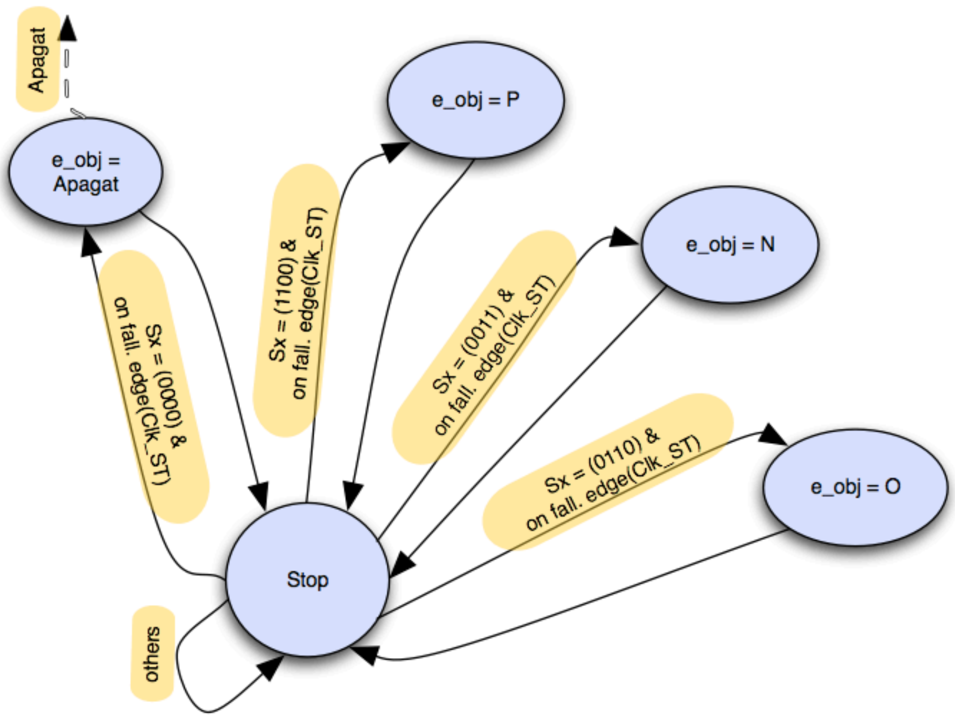

Apagat
e_obj = Apagat
e_obj = P
e_obj = N
e_obj = O
Stop
Sx = (0000) & on fall. edge(Clk_ST)
Sx = (1100) & on fall. edge(Clk_ST)
Sx = (0011) & on fall. edge(Clk_ST)
Sx = (0110) & on fall. edge(Clk_ST)
others

Fig. 12.10: *Selecció Estat Objectiu* Process

This process sets the value of the signal *e_obj* based on the input Sx[3 .. 0]. As shown in the diagram, refreshment of *e_obj* occurs each falling edge of *Clk_ST*.

Every *e_obj* updated with a different state respect to previous, process *Camí Transicions* begin the path to the goal state (*e_obj*), on the falling edge of *Clk_IN* immediately after.

12.5. Signals assignation to FPGA

FPGA Name	FPGA Pin	Signal	bytes "bits_32"	dSpace Signal
Inputs				
Engegada	129	Engegada	byte 4.6 (64)	IO30
Enviament	131	Enviament	byte 4.4 (16)	IO28
bit_0	138	San - BT[0]	byte 3	IO16
bit_1	149	Sap - BT[1]		IO17
bit_2	137	Sbn - BT[2]		IO18
bit_3	151	Sbp - BT[3]		IO19
bit_4	136	Scn - BT[4]		IO20
bit_5	152	Scp - BT[5]		IO21
bit_6	134	BT[6]		IO22
bit_7	153	BT[7]		IO23
bit_8	143	ST[0]	byte 2	IO8
bit_9	119	ST[1]		IO9
bit_10	142	ST[2]		IO10
bit_11	118	ST[3]		IO11
bit_12	141	ST[4]		IO12
bit_13	117	ST[5]		IO13
bit_14	139	ST[6]		IO14
bit_15	116	ST[7]		IO15
errora1	229	errora1		-
errora2	101	errora2		-
errora3	84	errora3		-
errora4	86	errora4		-
errorb1	87	errorb1		-
errorb2	88	errorb2		-
errorb3	94	errorb3		-
errorb4	95	errorb4		-
errorc1	97	errorc1		-
errorc2	98	errorc2		-
errorc3	99	errorc3		-
errorc4	100	errorc4		-

Tab. 12.11: Input Pin Assignment

FPGA Name	FPGA Pin	Signal	bytes "bits_32"	dSpace Signal
Outputs		Outputs		
clk_ST_out	148	clk_ST	byte 1.0	IO0
Sa1	64	Sa1		-
Sa2	67	Sa2		-
Sa3	65	Sa3		-
Sa4	66	Sa4		-
Sb1	56	Sb1		-
Sb2	62	Sb2		-
Sb3	61	Sb3		-
Sb4	63	Sb4		-
Sc1	51	Sc1		-
Sc2	55	Sc2		-
Sc3	53	Sc3		-
Sc4	54	Sc4		-
error_trans	161	error_trans		INT2
clk_ST_int	162	clk_ST		INT3
error_drivers	163	error_drivers		INT4

Tab. 12.12: Output Pin Assignment

12.6. FPGA test

Now the functioning of the FPGA programming is show. For obtaining the result, an oscilloscope has been connected directly to the corresponding outputs of the FPGA, 4 pins that connect the FPGA to IGBT drivers. These 4 signals are connected to a … oscilloscope with digital state function, that allow to see the transitions of the IGBTs.

Also the *enviament* signal is connected to the oscilloscope, because this signal has been used like trigger signal. When the oscilloscope in "single condition" running read a change of level in the *enviament* signal, makes a read of all 4 signals.

The control of the system has been realized with a special Simulink program and a Control Desk user interface dedicated. Al possible transition are tested.

Next Simulink diagram, Control Desk interface and all the results are shown.

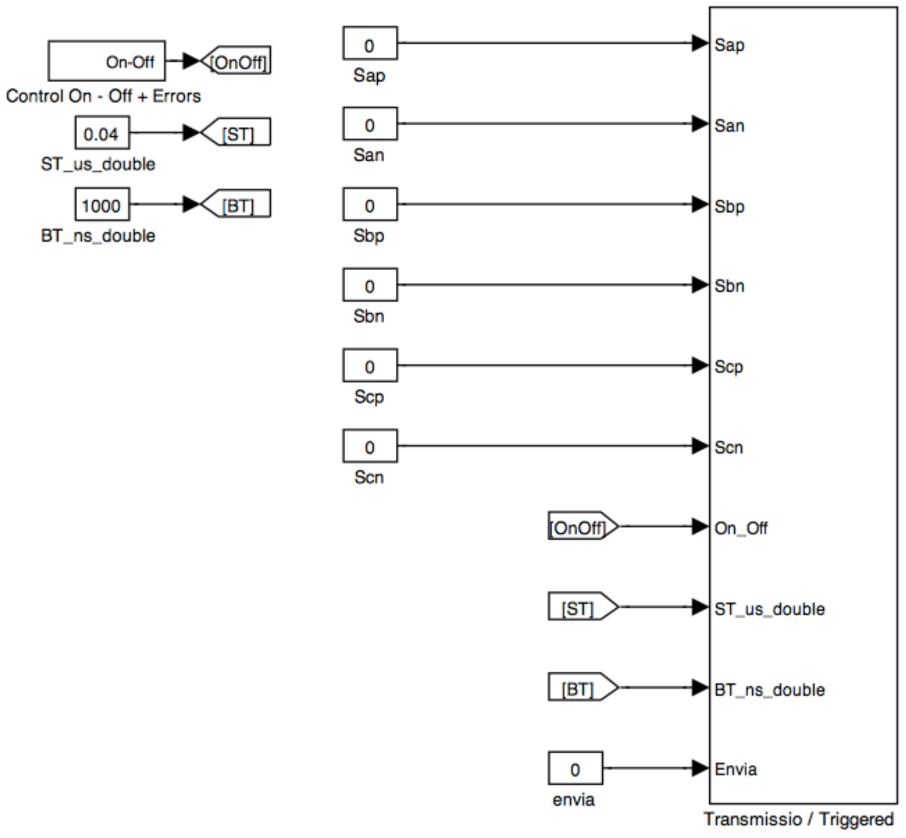


Fig. 12.11: Test Salida Simulink diagram. Signal Sax, Sbx, Scx and envía are connected to related button in the Control Desk interface.

In next fig channel 1 (on the top) in Envia signal, and on he bottom we have from P to N 4 IGBT driver's signals of one branch, respectively of S1, S22, S11 and S2 of fig. 4.4 of the memory of this project.

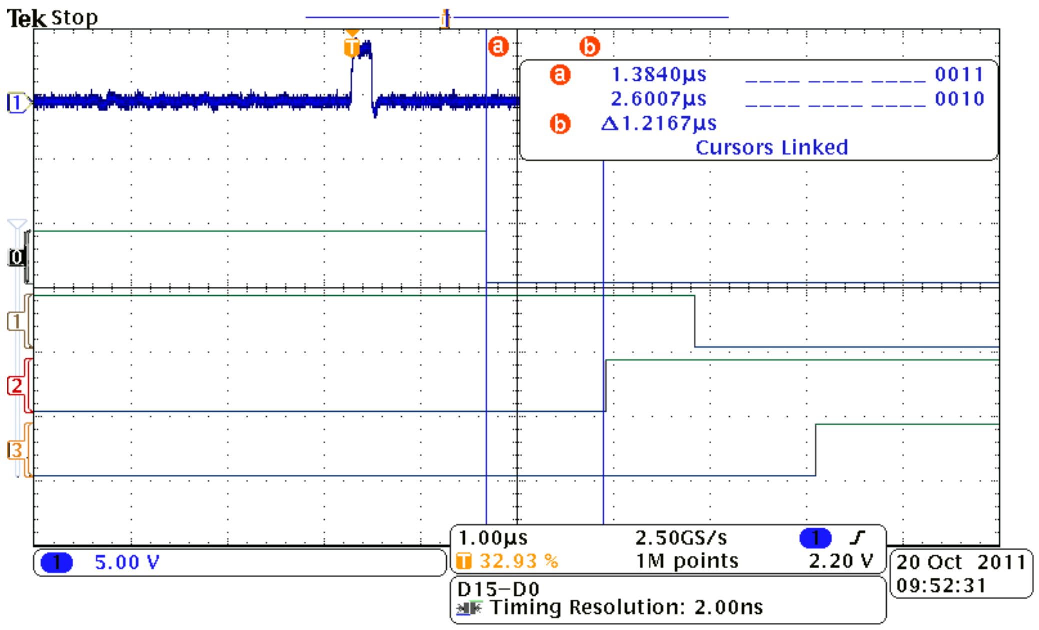


Fig. 12.12: P to N transition with Blanking Time= 1 µs

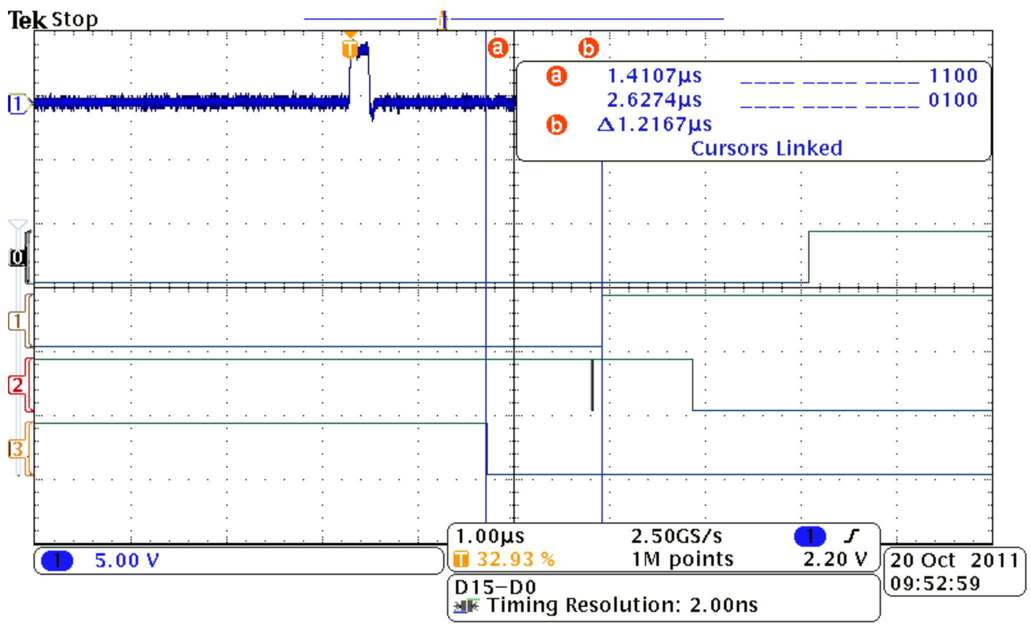


Fig. 12.13: N to P transition with Blanking time = 1 µs

13. Control Desk

For real time monitoring of the converter functioning and have the possibility of changing parameters, a Control Desk user interface has been created. Next figure shows the aspect of this realization. Elements that form a part of the system are directly linked to Simulink constants executed in the dSpace like discussed in appendix B.

No se puede mostrar la imagen. Puede que su equipo no tenga suficiente memoria para abrir la imagen o que ésta esté dañada. Reinicie el equipo y, a continuación, abra el archivo de nuevo. Si sigue apareciendo la x roja, puede que tenga que borrar la imagen e insertarla de nuevo.

Fig. 13.1: ControlDesk User interface

14. Simulink Simulation Model

Different Simulink models have been created for the realization of simulations of the system. These were necessary for test the functioning before than run the program directly in the real converter, to avoid the malfunctioning and possible damaging of the laboratory instruments.

The goal was to obtain similar values to measured in the inverter, allow to a later analysis of anomalies in experimental tests.

14.1. Model

Next figure shows Simulink diagram of simulation's model.

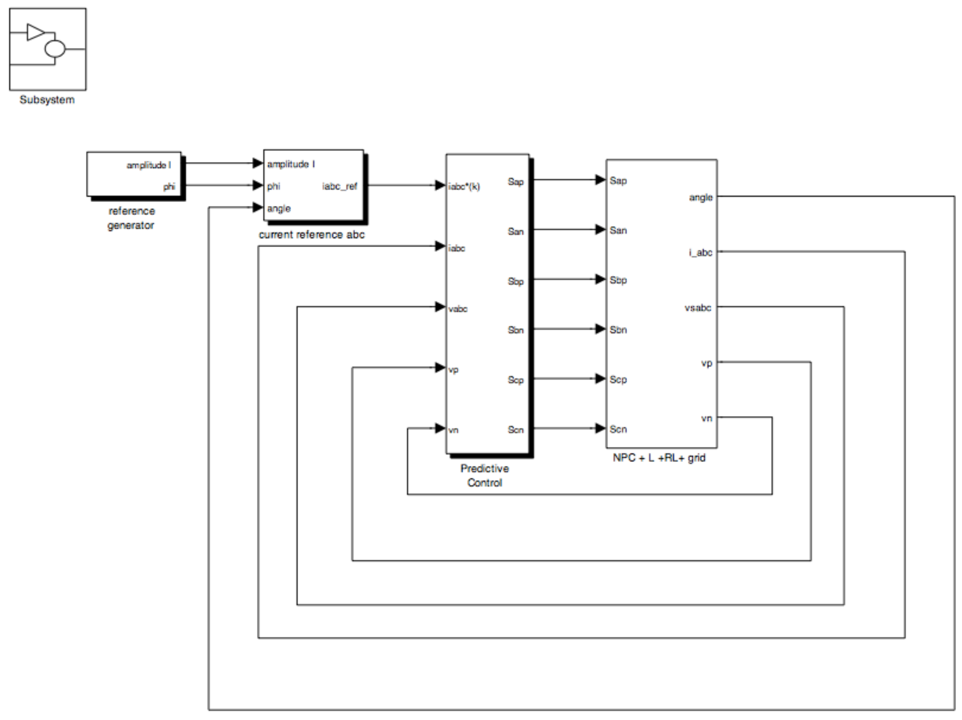


Fig. 14.1: Simulation Model

Predictive Control block is the same than the experimental diagram, with the only difference of a Sample Time Simulink block in every input and every output. These are necessary to synchronize the reading and elaboration of all the signals, that now are sampled every 100 μs.

The block that realizes the analysis of measured values in the inverter is no longer present, due to the realization of the load model, *NPC+L+RL+grid* block.

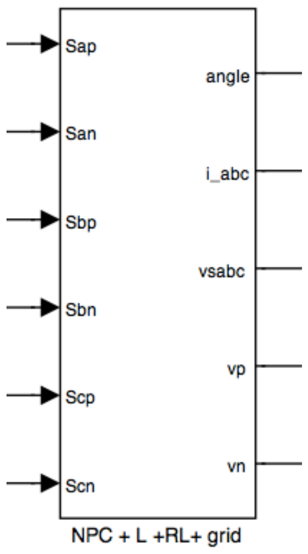


Fig. 14.2: NPC+L+RL+grid block

The internal of this block is shown in next figure.

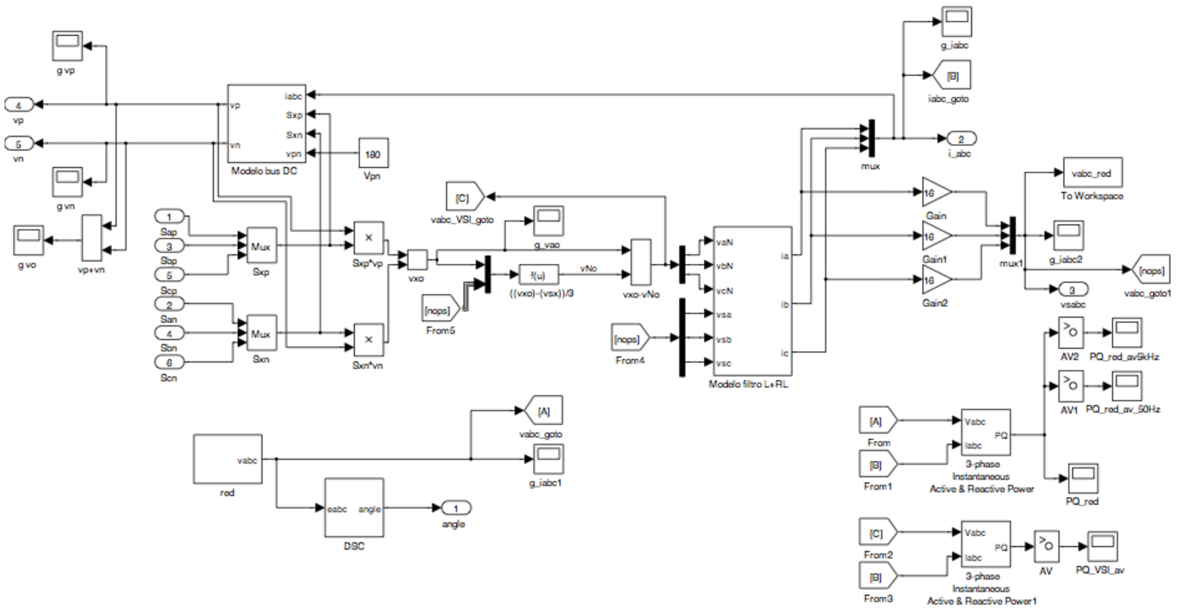


Fig. 14.3: NPC+L+RL+grid block inside for R load simulation

This model simulate the condition of the system connected to a R Load of 16 Ω in every phase. This is realized from blocks present in next image.

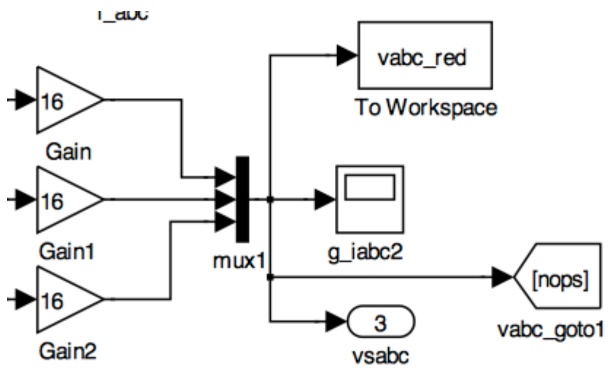


Fig. 14.4: R load simulation block

This system takes values of i_a , i_b , i_c and multiply these for a gain of 16, obtaining the 2 voltages of every branch.

Also model of the R+RL filter and DC link are present, with values like described in cap.7 of the memory of this project.

Next figure shows model for the simulation with network load.

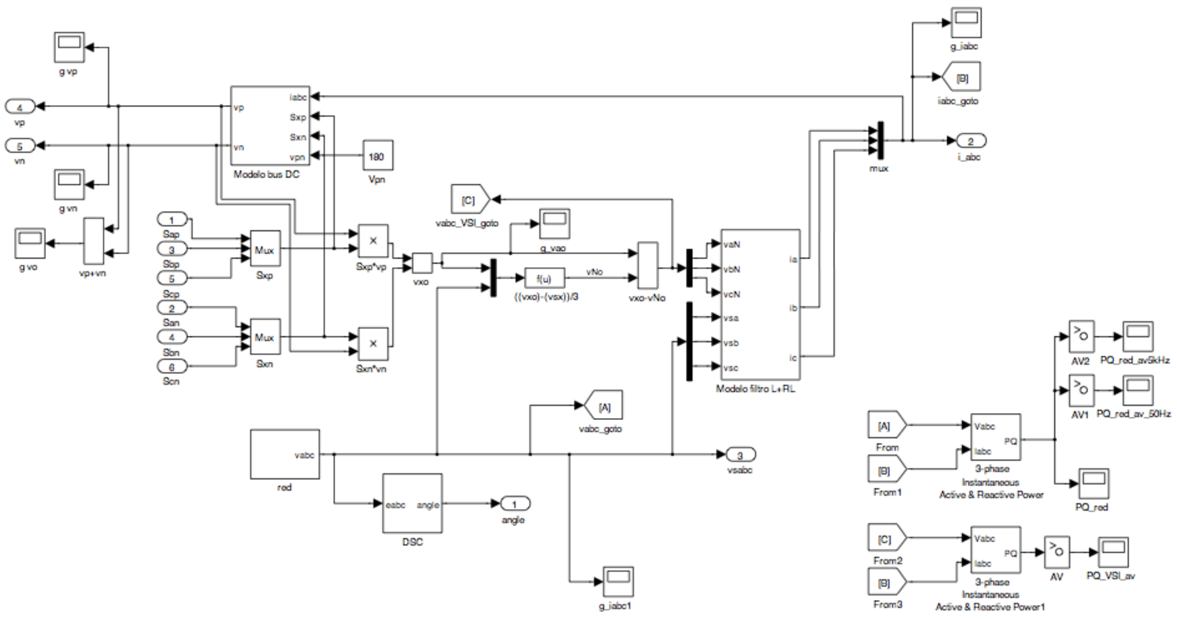


Fig. 14.5: NPC+L+RL+grid block inside for network load simulation

And next figure shows the system used for the network load.

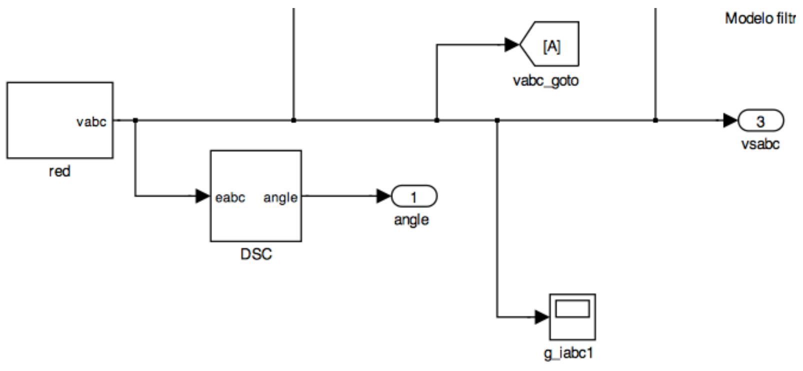


Fig. 14.6: Network Load model

Red block create a 3-phase currents system directly connected to the vsabc output. Values of amplitude and phase are described in cap. 7 of the memory of this project.

15. Economic Study

This Appendix shows a summary of the economic expenses related to this project. Depending on the nature and tasks performed, these costs can be classified into the following concepts:

- Human resources
- Equipment cushioning
- Various expenses

Material costs are not considered, since for this project were not made specific purchases.

Next sections described the contents of every category.

15.1. Human resources

Under the concept of human resources, the work done by the team of people who has had a role in this project is considered. This team is composed of a group of senior engineers and a junior engineer, performing the following tasks:

- **Senior engineers**: formed by project managers, are responsible for the project management, recommending appropriate lines of work.
- **Junior Engineer**: person who develops the project, responsible for technical and administrative tasks.

The cost related to human resources is shown in next table.

CONCEPT	Necessary Hours	Hour Cost	Total Cost
Senior Engineer	300 hours	60 euros/h	18000 euros
Junior Engineer	1280 hours	20 euros/h	25600 euros

Tab. 15.1: Human Resources Cost

15.2. Equipment Cushioning

In this section we have considered the costs of depreciation of equipment and software necessary to carry out the project. The repayment model is linear and is based on hours of use of each component.

CONCEPT	Price	Life	Maintenance cost	Cushioning (€/h)	Hours	HW Cost
3-level converter	13.200 €	13,200 h	8%	1,08	180	194,40 €
UP2 Altera Board	180 €	5,000 h	8%	0,04	350	14,00 €
dSpace 1103	10.210 €	10,000 h	8%	1,1	350	385,00 €
Tektronix MSO3000 series oscilloscope	10.500 €	10,000 h	8%	1,13	120	135,60 €
PC Pentium II	400 €	15,000 h	8%	0,03	450	13,50 €
Notebook HP Pavilion DV5000	1.200 €	15,000 h	8%	0,09	900	81,00 €
Resistive load IER-E23612	636 €	5,000 h	8%	0,14	180	25,20 €
Inductive Filter	190 €	3,000 h	8%	0,07	180	12,60 €
Supply DC HP 6030A 1000 W	4.050 €	10,000 h	8%	0,44	180	79,20 €
Supply AC/DC California Instruments 5001iX, 5000 VA	16.090 €	10,000 h	8%	1,74	90	156,60 €
					TOTAL	1.097,10 €

Tab. 15.2: Hardware Cushioning

CONCEPT	Price	Life	Maintenance cost	Cushioning (€/h)	Hours	HW Cost
Windows 98	300 €	15,000 h	8%	0,02 €	450	9,72 €
Windows XP i Microsoft Office	300 €	15,000 h	8%	0,02 €	900	19,44 €
MATLAB + Simulink	450 €	15,000 h	8%	0,03 €	1350	43,74 €
Maxplus II Baseline 10.2	0 €	15,000 h	8%	0 €	450	0,00 €
dSpace Control Desk	0 €	15,000 h	8%	0 €	180	0,00 €
					TOTAL	72,90 €

Tab. 15.3: Software Cushioning

15.3. Various Expenses

An added cost of 19% (rate apply by UPC to his projects) in considered on the sum of total costs, like electricity and water wastes during project's realization period.

So, total cost for various expense srise up to 8506,3 €.

15.4. Total Cost of the Project

The sum of all costs give us the total cost of this project.

CONCEPT	COST
Human Resources	€ 43.600,00
Equipment Cushioning	€ 1.170,00
Various Expenses	€ 8.506,30
TOTAL	€ 53.276,30

Tab. 15.4: Total cost of the Project

16. Environmental Study

Environmental impact of this project is practically zero, because is composed from theoretical studies and a laboratory tests without any kind of emission.

We can consider like environmental impact the fact that the equipment used have a lifetime, and in the end of this life need to be recycled.

Is also possible to evaluate CO2 emissions due to the use of energy for the supply of all the equipment during tests. The way of energy creation can modify the amount of CO2 generated, so renewable energy creates less waste than a fossil station.

For sure the material in this project can help the environment, because of the suitability in wind power generation plants, which generate electricity by a renewable way, reducing the level of environmental pollution.

17. Equipment Data Sheets

This Appendix shows data sheets of components used in this project.

17.1. Artesyn Technology NPL 65 Supply Board

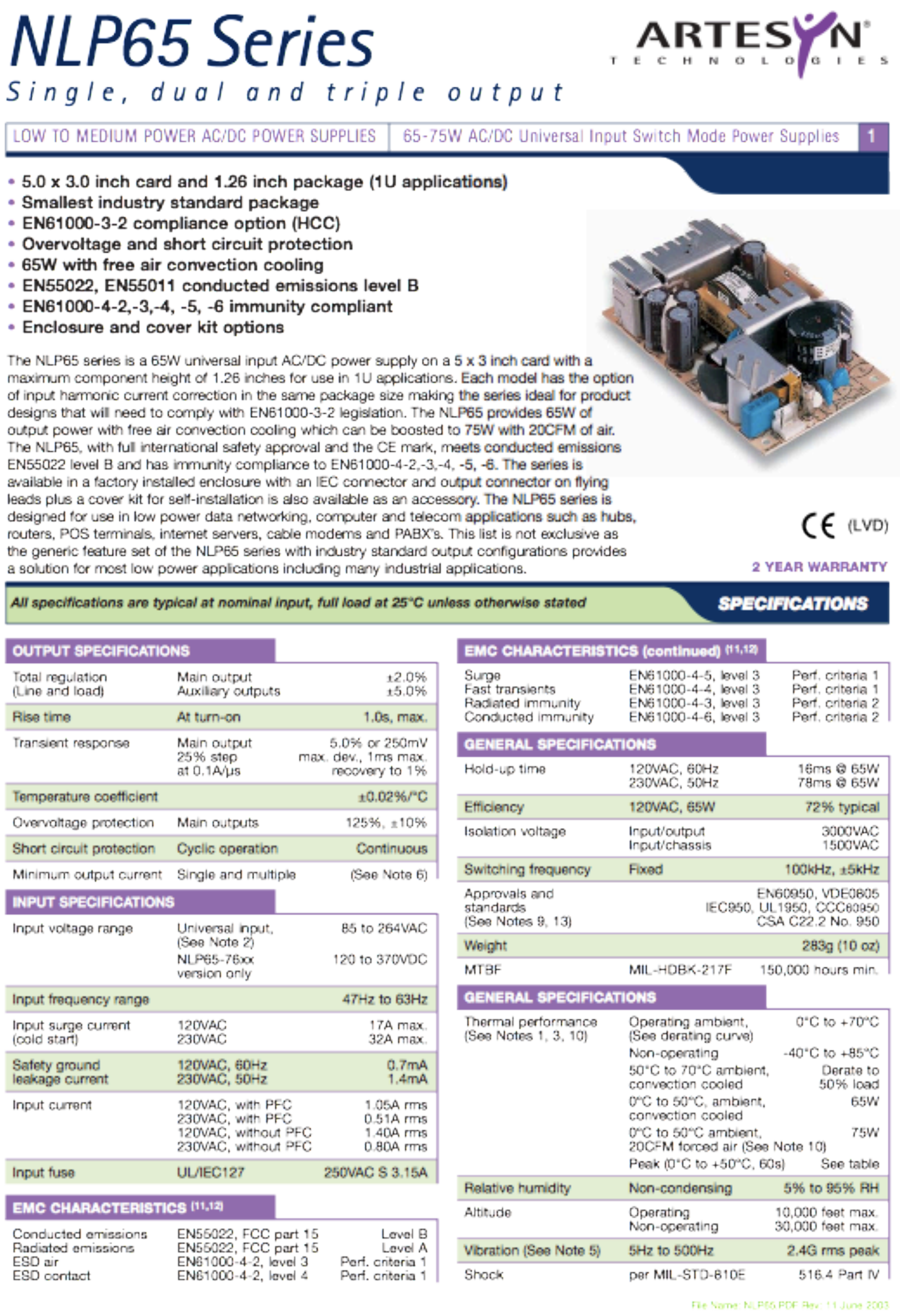
17.2. dSpace 1103 Board

17.3. Altera EPF10K70 FPGA

17.4. Semikron SKM100 IGBT

17.5. Semikron SKHI 10 Drivers

17.1. Artesyn Technology NPL 65 Supply Board



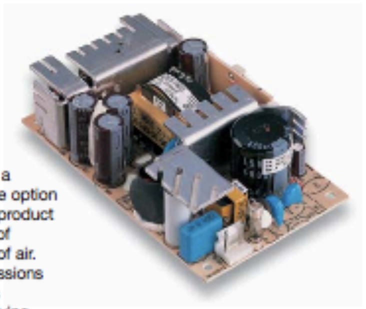
NLP65 Series

Single, dual and triple output

ARTESYN TECHNOLOGIES

LOW TO MEDIUM POWER AC/DC POWER SUPPLIES | 65-75W AC/DC Universal Input Switch Mode Power Supplies | 1

- 5.0 x 3.0 inch card and 1.26 inch package (1U applications)
- Smallest industry standard package
- EN61000-3-2 compliance option (HCC)
- Overvoltage and short circuit protection
- 65W with free air convection cooling
- EN55022, EN55011 conducted emissions level B
- EN61000-4-2,-3,-4, -5, -6 immunity compliant
- Enclosure and cover kit options



The NLP65 series is a 65W universal input AC/DC power supply on a 5 x 3 inch card with a maximum component height of 1.26 inches for use in 1U applications. Each model has the option of input harmonic current correction in the same package size making the series ideal for product designs that will need to comply with EN61000-3-2 legislation. The NLP65 provides 65W of output power with free air convection cooling which can be boosted to 75W with 20CFM of air. The NLP65, with full international safety approval and the CE mark, meets conducted emissions EN55022 level B and has immunity compliance to EN61000-4-2,-3,-4, -5, -6. The series is available in a factory installed enclosure with an IEC connector and output connector on flying leads plus a cover kit for self-installation is also available as an accessory. The NLP65 series is designed for use in low power data networking, computer and telecom applications such as hubs, routers, POS terminals, internet servers, cable modems and PABX's. This list is not exclusive as the generic feature set of the NLP65 series with industry standard output configurations provides a solution for most low power applications including many industrial applications.

CE (LVD)

2 YEAR WARRANTY

All specifications are typical at nominal input, full load at 25°C unless otherwise stated **SPECIFICATIONS**

OUTPUT SPECIFICATIONS

Total regulation (Line and load)	Main output Auxiliary outputs	±2.0% ±5.0%
Rise time	At turn-on	1.0s, max.
Transient response	Main output 25% step at 0.1A/µs	5.0% or 250mV max. dev., 1ms max. recovery to 1%
Temperature coefficient		±0.02%/°C
Overvoltage protection	Main outputs	125%, ±10%
Short circuit protection	Cyclic operation	Continuous
Minimum output current	Single and multiple	(See Note 6)

INPUT SPECIFICATIONS

Input voltage range	Universal input, (See Note 2) NLP65-76xx version only	85 to 264VAC 120 to 370VDC
Input frequency range		47Hz to 63Hz
Input surge current (cold start)	120VAC 230VAC	17A max. 32A max.
Safety ground leakage current	120VAC, 60Hz 230VAC, 50Hz	0.7mA 1.4mA
Input current	120VAC, with PFC 230VAC, with PFC 120VAC, without PFC 230VAC, without PFC	1.05A rms 0.51A rms 1.40A rms 0.80A rms
Input fuse	UL/IEC127	250VAC S 3.15A

EMC CHARACTERISTICS (11,12)

Conducted emissions	EN55022, FCC part 15	Level B
Radiated emissions	EN55022, FCC part 15	Level A
ESD air	EN61000-4-2, level 3	Perf. criteria 1
ESD contact	EN61000-4-2, level 4	Perf. criteria 1

EMC CHARACTERISTICS (continued) (11,12)

Surge	EN61000-4-5, level 3	Perf. criteria 1
Fast transients	EN61000-4-4, level 3	Perf. criteria 1
Radiated immunity	EN61000-4-3, level 3	Perf. criteria 2
Conducted immunity	EN61000-4-6, level 3	Perf. criteria 2

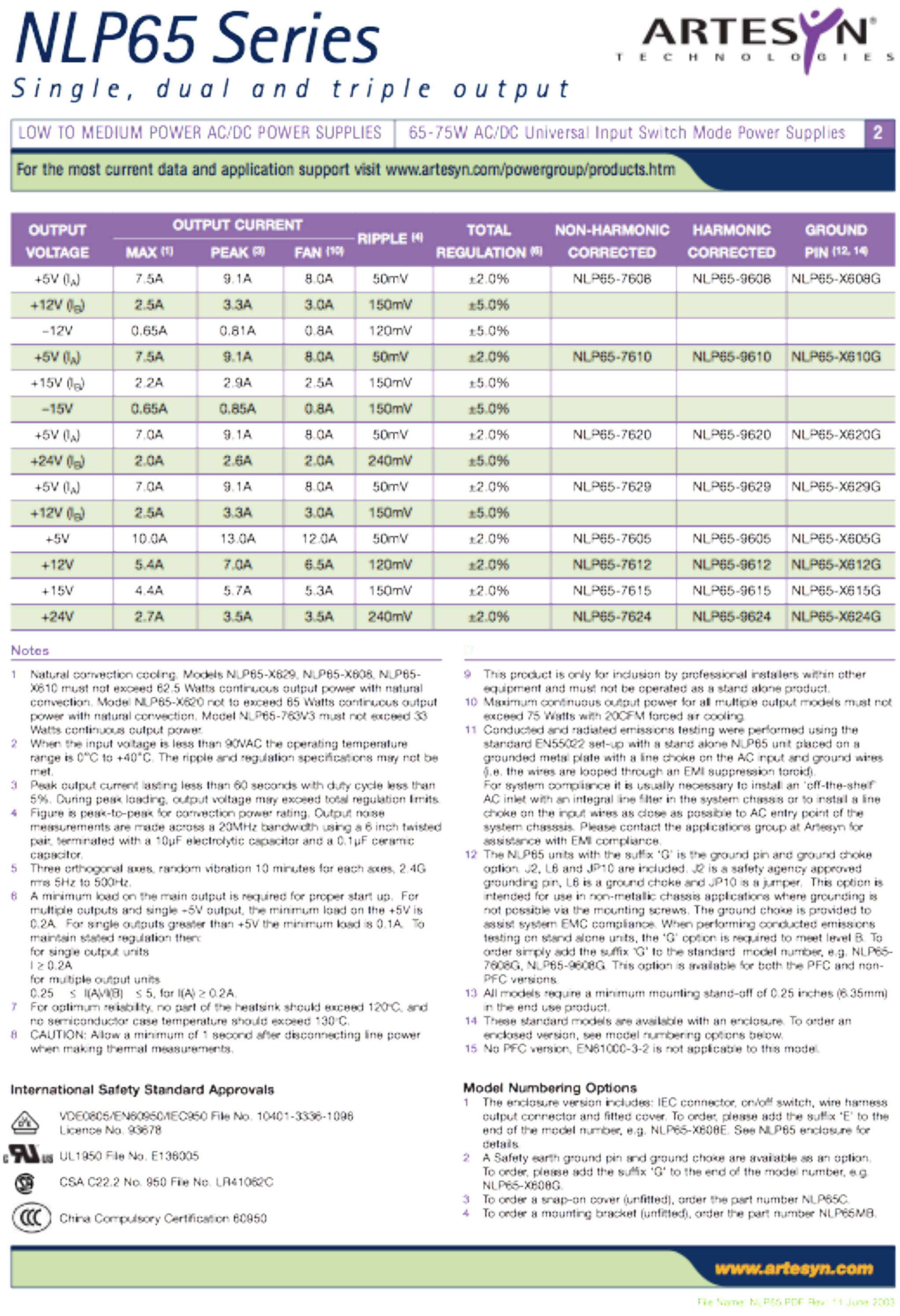
GENERAL SPECIFICATIONS

Hold-up time	120VAC, 60Hz 230VAC, 50Hz	16ms @ 65W 78ms @ 65W
Efficiency	120VAC, 65W	72% typical
Isolation voltage	Input/output Input/chassis	3000VAC 1500VAC
Switching frequency	Fixed	100kHz, ±5kHz
Approvals and standards (See Notes 9, 13)		EN60950, VDE0805 IEC950, UL1950, CCC60950 CSA C22.2 No. 950
Weight		283g (10 oz)
MTBF	MIL-HDBK-217F	150,000 hours min.

GENERAL SPECIFICATIONS

Thermal performance (See Notes 1, 3, 10)	Operating ambient, (See derating curve)	0°C to +70°C
	Non-operating	-40°C to +85°C
	50°C to 70°C ambient, convection cooled	Derate to 50% load
	0°C to 50°C, ambient, convection cooled	65W
	0°C to 50°C ambient, 20CFM forced air (See Note 10)	75W
	Peak (0°C to +50°C, 60s)	See table
Relative humidity	Non-condensing	5% to 95% RH
Altitude	Operating Non-operating	10,000 feet max. 30,000 feet max.
Vibration (See Note 5)	5Hz to 500Hz	2.4G rms peak
Shock	per MIL-STD-810E	516.4 Part IV

File Name: NLP65 PDF Rev: 11 June 2003



NLP65 Series

Single, dual and triple output

ARTESYN TECHNOLOGIES

LOW TO MEDIUM POWER AC/DC POWER SUPPLIES | 65-75W AC/DC Universal Input Switch Mode Power Supplies | 2

For the most current data and application support visit www.artesyn.com/powergroup/products.htm

OUTPUT VOLTAGE	OUTPUT CURRENT			RIPPLE [4]	TOTAL REGULATION [6]	NON-HARMONIC CORRECTED	HARMONIC CORRECTED	GROUND PIN [12, 14]
	MAX [1]	PEAK [3]	FAN [10]					
+5V (I_A)	7.5A	9.1A	8.0A	50mV	±2.0%	NLP65-7608	NLP65-9608	NLP65-X608G
+12V (I_B)	2.5A	3.3A	3.0A	150mV	±5.0%			
−12V	0.65A	0.81A	0.8A	120mV	±5.0%			
+5V (I_A)	7.5A	9.1A	8.0A	50mV	±2.0%	NLP65-7610	NLP65-9610	NLP65-X610G
+15V (I_B)	2.2A	2.9A	2.5A	150mV	±5.0%			
−15V	0.65A	0.85A	0.8A	150mV	±5.0%			
+5V (I_A)	7.0A	9.1A	8.0A	50mV	±2.0%	NLP65-7620	NLP65-9620	NLP65-X620G
+24V (I_B)	2.0A	2.6A	2.0A	240mV	±5.0%			
+5V (I_A)	7.0A	9.1A	8.0A	50mV	±2.0%	NLP65-7629	NLP65-9629	NLP65-X629G
+12V (I_B)	2.5A	3.3A	3.0A	150mV	±5.0%			
+5V	10.0A	13.0A	12.0A	50mV	±2.0%	NLP65-7605	NLP65-9605	NLP65-X605G
+12V	5.4A	7.0A	6.5A	120mV	±2.0%	NLP65-7612	NLP65-9612	NLP65-X612G
+15V	4.4A	5.7A	5.3A	150mV	±2.0%	NLP65-7615	NLP65-9615	NLP65-X615G
+24V	2.7A	3.5A	3.5A	240mV	±2.0%	NLP65-7624	NLP65-9624	NLP65-X624G

Notes

1 Natural convection cooling. Models NLP65-X629, NLP65-X608, NLP65-X610 must not exceed 62.5 Watts continuous output power with natural convection. Model NLP65-X620 not to exceed 65 Watts continuous output power with natural convection. Model NLP65-763V3 must not exceed 33 Watts continuous output power.

2 When the input voltage is less than 90VAC the operating temperature range is 0°C to +40°C. The ripple and regulation specifications may not be met.

3 Peak output current lasting less than 60 seconds with duty cycle less than 5%. During peak loading, output voltage may exceed total regulation limits.

4 Figure is peak-to-peak for convection power rating. Output noise measurements are made across a 20MHz bandwidth using a 6 inch twisted pair, terminated with a 10µF electrolytic capacitor and a 0.1µF ceramic capacitor.

5 Three orthogonal axes, random vibration 10 minutes for each axes, 2.4G rms 5Hz to 500Hz.

6 A minimum load on the main output is required for proper start up. For multiple outputs and single +5V output, the minimum load on the +5V is 0.2A. For single outputs greater than +5V the minimum load is 0.1A. To maintain stated regulation then:
for single output units
$I \geq 0.2A$
for multiple output units
$0.25 \leq I(A)/I(B) \leq 5$, for $I(A) \geq 0.2A$.

7 For optimum reliability, no part of the heatsink should exceed 120°C, and no semiconductor case temperature should exceed 130°C.

8 CAUTION: Allow a minimum of 1 second after disconnecting line power when making thermal measurements.

9 This product is only for inclusion by professional installers within other equipment and must not be operated as a stand alone product.

10 Maximum continuous output power for all multiple output models must not exceed 75 Watts with 20CFM forced air cooling.

11 Conducted and radiated emissions testing were performed using the standard EN55022 set-up with a stand alone NLP65 unit placed on a grounded metal plate with a line choke on the AC input and ground wires (i.e. the wires are looped through an EMI suppression toroid).
For system compliance it is usually necessary to install an 'off-the-shelf' AC inlet with an integral line filter in the system chassis or to install a line choke on the input wires as close as possible to AC entry point of the system chasssis. Please contact the applications group at Artesyn for assistance with EMI compliance.

12 The NLP65 units with the suffix 'G' is the ground pin and ground choke option. J2, L6 and JP10 are included. J2 is a safety agency approved grounding pin, L6 is a ground choke and JP10 is a jumper. This option is intended for use in non-metallic chassis applications where grounding is not possible via the mounting screws. The ground choke is provided to assist system EMC compliance. When performing conducted emissions testing on stand alone units, the 'G' option is required to meet level B. To order simply add the suffix 'G' to the standard model number, e.g. NLP65-7608G, NLP65-9608G. This option is available for both the PFC and non-PFC versions.

13 All models require a minimum mounting stand-off of 0.25 inches (6.35mm) in the end use product.

14 These standard models are available with an enclosure. To order an enclosed version, see model numbering options below.

15 No PFC version, EN61000-3-2 is not applicable to this model.

International Safety Standard Approvals

VDE0805/EN60950/IEC950 File No. 10401-3336-1096
Licence No. 93678

UL1950 File No. E136005

CSA C22.2 No. 950 File No. LR41062C

China Compulsory Certification 60950

Model Numbering Options

1 The enclosure version includes: IEC connector, on/off switch, wire harness output connector and fitted cover. To order, please add the suffix 'E' to the end of the model number, e.g. NLP65-X608E. See NLP65 enclosure for details.

2 A Safety earth ground pin and ground choke are available as an option. To order, please add the suffix 'G' to the end of the model number, e.g. NLP65-X608G.

3 To order a snap-on cover (unfitted), order the part number NLP65C.

4 To order a mounting bracket (unfitted), order the part number NLP65MB.

www.artesyn.com

File Name: NLP65 PDF Rev: 11 June 2003

NLP65 Series

Single, dual and triple output

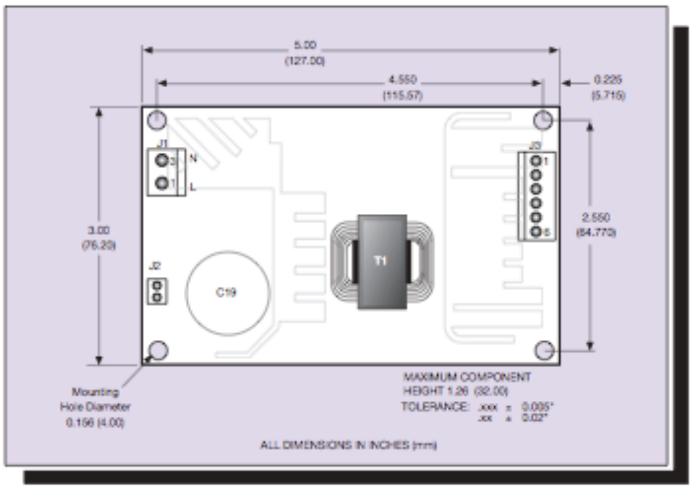
ARTESYN TECHNOLOGIES

LOW TO MEDIUM POWER AC/DC POWER SUPPLIES | 65-75W AC/DC Universal Input Switch Mode Power Supplies | 3

For the most current data and application support visit www.artesyn.com/powergroup/products.htm

Mechanical Notes

A All dimensions are in inches (mm).



INPUT PIN CONNECTIONS	
J1	
Pin 1	AC Line
Pin 2	No Pin
Pin 3	AC Neutral
J2 (ON 'G' SUFFIX ONLY)	
Pin 1	Safety Ground

OUTPUT PIN CONNECTIONS				
J3	**SINGLE -XX05 ONLY**	**SINGLE**	**DUAL**	**TRIPLE**
Pin 1	V (A)	No Connection	V (B)	V (B)
Pin 2	V (A)	V (A)	V (A)	V (A)
Pin 3	V (A)	V (A)	V (A)	V (A)
Pin 4	Return	Return	Return	Return
Pin 5	Return	Return	Return	Return
Pin 6	Return	No Connection	N/C	V (C)

Input and output connectors

AC (J1) connector type
Molex 26-60-4030 type.

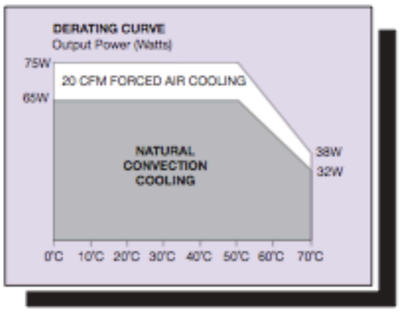
DC (J3) connector type
Molex 26-60-4060 type.

Mating connectors

AC (J1) mating connector type
Molex 09-50-3031 or equivalent with Molex 08-50-0105 or equivalent crimp terminals.

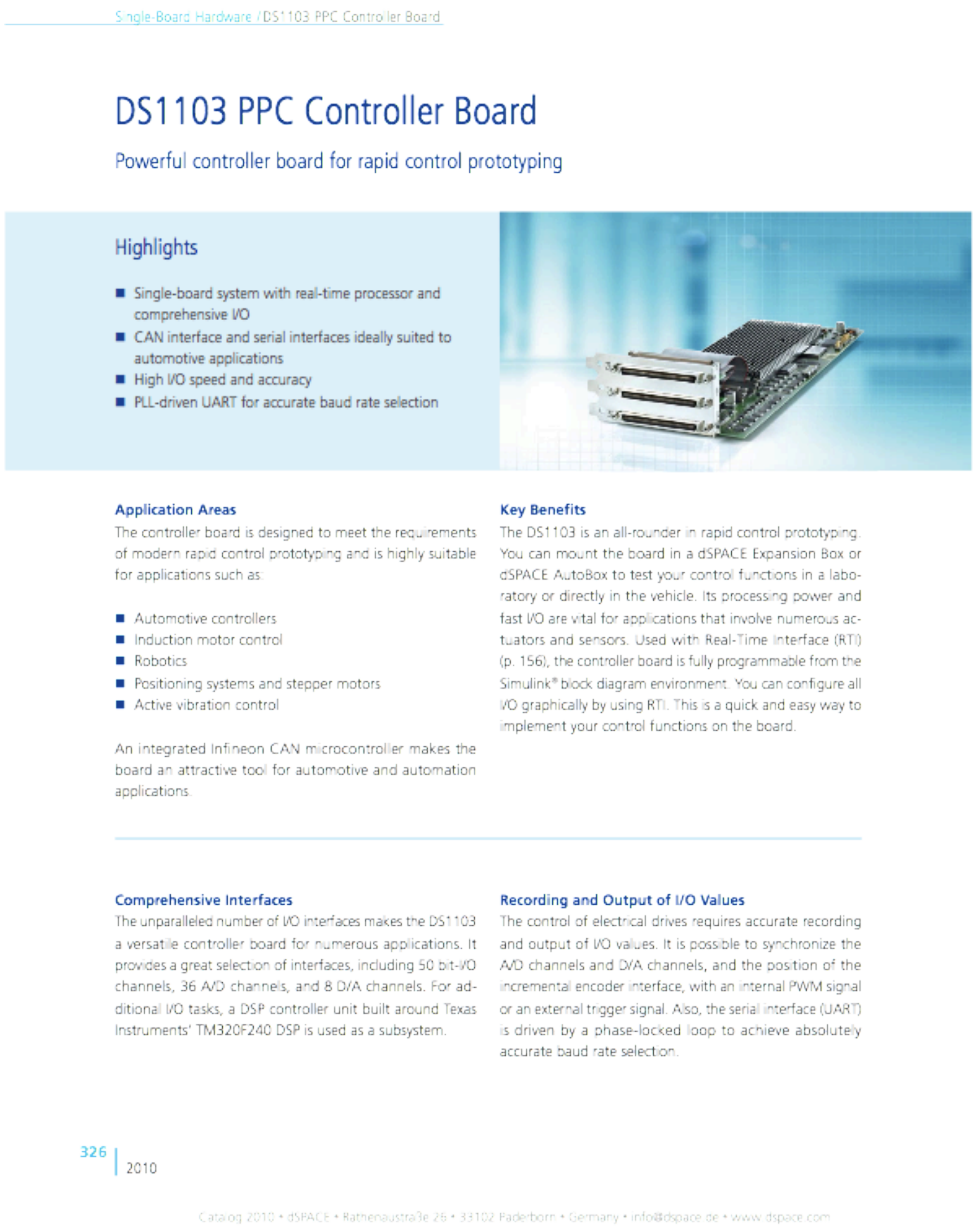
DC (J3) mating connector type
Molex 09-50-3061 with Molex 2478 phosphor bronze crimp terminals or equivalent.

Note: The input and output connectors are the same as those used on NFS40, NFN40, NAL40, NAN40 and NLP40.



www.artesyn.com

17.2. dSpace 1103 Board



Single-Board Hardware / DS1103 PPC Controller Board

DS1103 PPC Controller Board

Powerful controller board for rapid control prototyping

Highlights

- Single-board system with real-time processor and comprehensive I/O
- CAN interface and serial interfaces ideally suited to automotive applications
- High I/O speed and accuracy
- PLL-driven UART for accurate baud rate selection

Application Areas

The controller board is designed to meet the requirements of modern rapid control prototyping and is highly suitable for applications such as:

- Automotive controllers
- Induction motor control
- Robotics
- Positioning systems and stepper motors
- Active vibration control

An integrated Infineon CAN microcontroller makes the board an attractive tool for automotive and automation applications.

Key Benefits

The DS1103 is an all-rounder in rapid control prototyping. You can mount the board in a dSPACE Expansion Box or dSPACE AutoBox to test your control functions in a laboratory or directly in the vehicle. Its processing power and fast I/O are vital for applications that involve numerous actuators and sensors. Used with Real-Time Interface (RTI) (p. 156), the controller board is fully programmable from the Simulink® block diagram environment. You can configure all I/O graphically by using RTI. This is a quick and easy way to implement your control functions on the board.

Comprehensive Interfaces

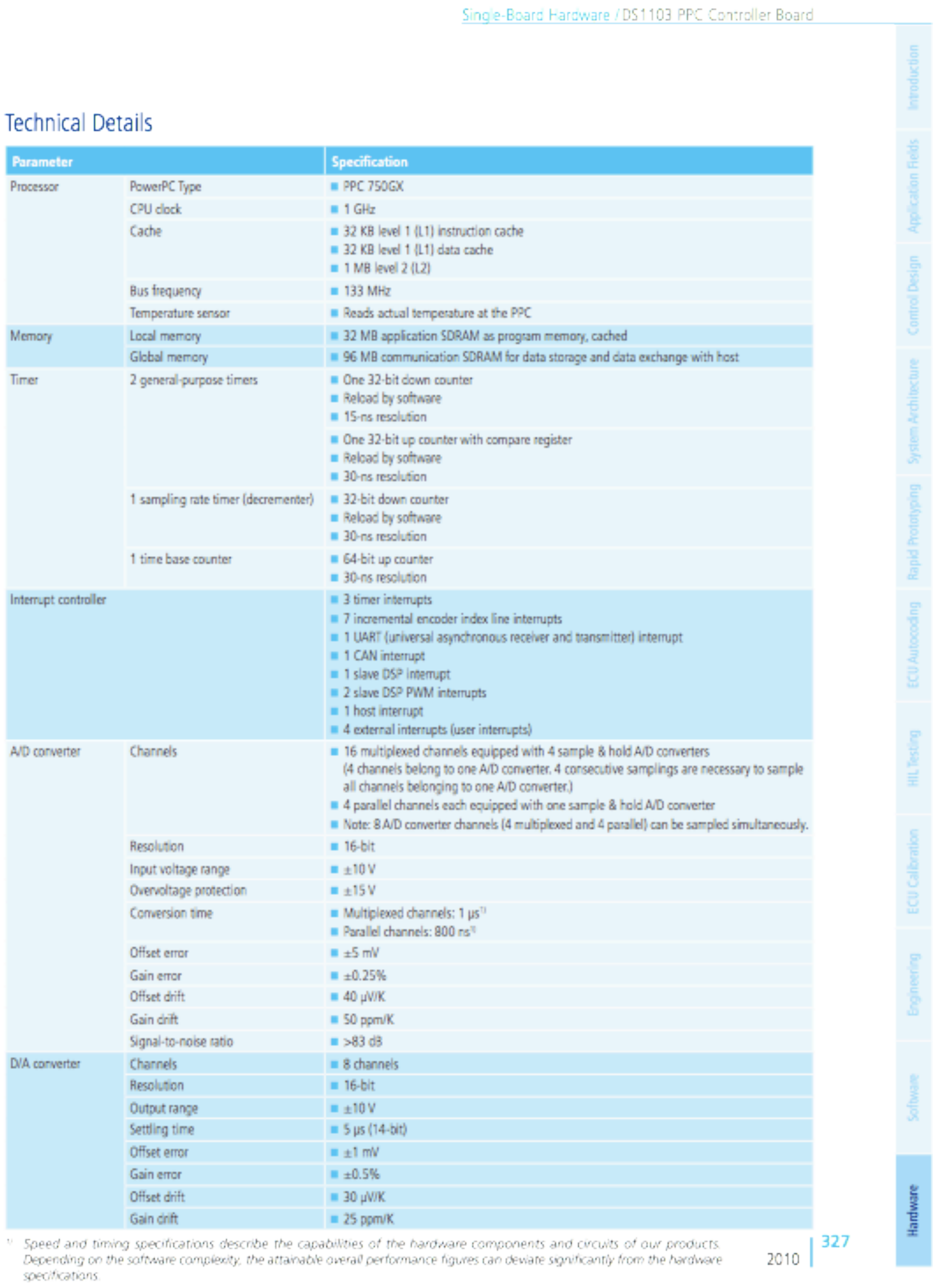
The unparalleled number of I/O interfaces makes the DS1103 a versatile controller board for numerous applications. It provides a great selection of interfaces, including 50 bit-I/O channels, 36 A/D channels, and 8 D/A channels. For additional I/O tasks, a DSP controller unit built around Texas Instruments' TM320F240 DSP is used as a subsystem.

Recording and Output of I/O Values

The control of electrical drives requires accurate recording and output of I/O values. It is possible to synchronize the A/D channels and D/A channels, and the position of the incremental encoder interface, with an internal PWM signal or an external trigger signal. Also, the serial interface (UART) is driven by a phase-locked loop to achieve absolutely accurate baud rate selection.

326 | 2010

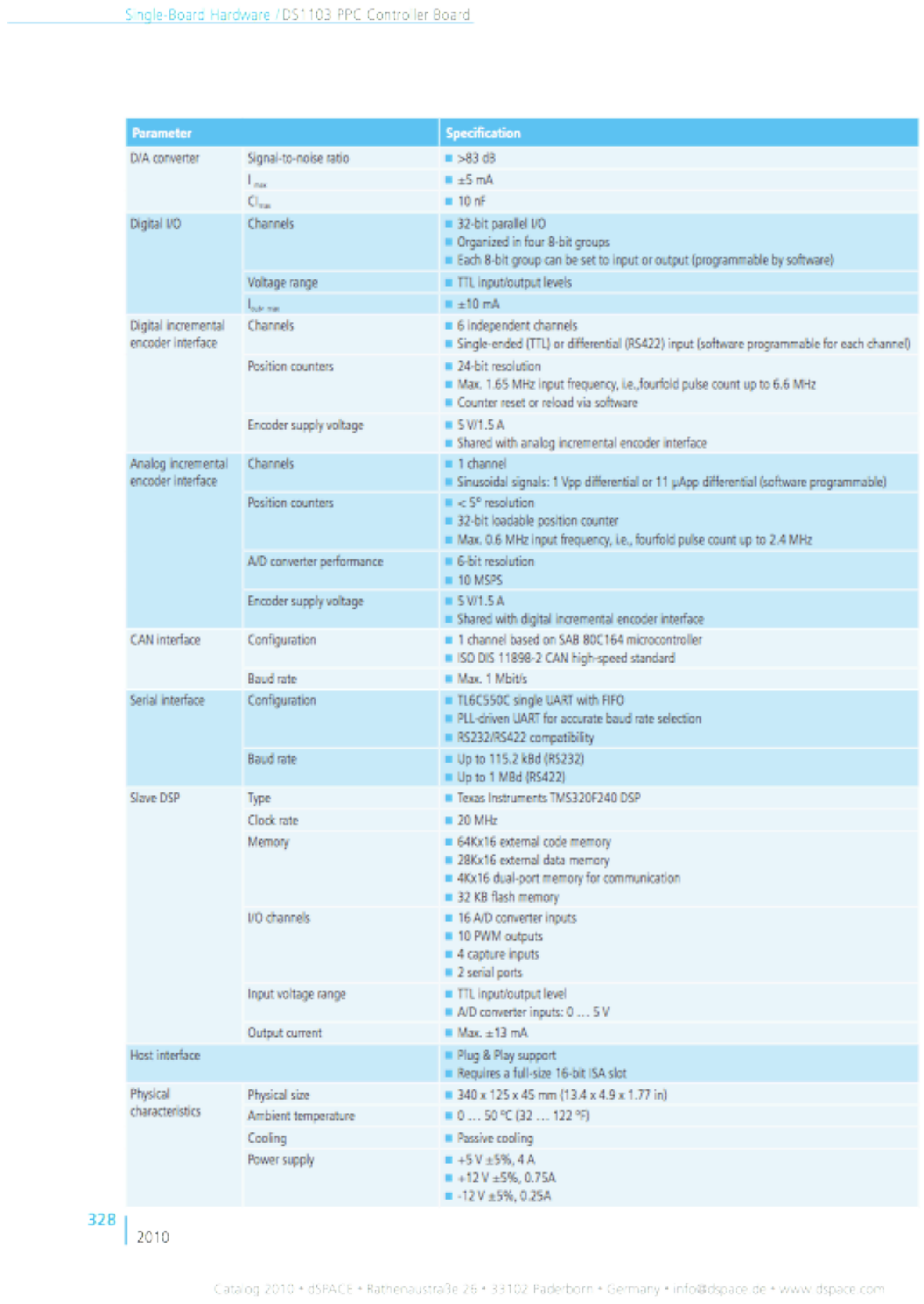
Catalog 2010 • dSPACE • Rathenaustraße 26 • 33102 Paderborn • Germany • info@dspace.de • www.dspace.com



Technical Details

Parameter		Specification
Processor	PowerPC Type	■ PPC 750GX
	CPU clock	■ 1 GHz
	Cache	■ 32 KB level 1 (L1) instruction cache ■ 32 KB level 1 (L1) data cache ■ 1 MB level 2 (L2)
	Bus frequency	■ 133 MHz
	Temperature sensor	■ Reads actual temperature at the PPC
Memory	Local memory	■ 32 MB application SDRAM as program memory, cached
	Global memory	■ 96 MB communication SDRAM for data storage and data exchange with host
Timer	2 general-purpose timers	■ One 32-bit down counter ■ Reload by software ■ 15-ns resolution
		■ One 32-bit up counter with compare register ■ Reload by software ■ 30-ns resolution
	1 sampling rate timer (decrementer)	■ 32-bit down counter ■ Reload by software ■ 30-ns resolution
	1 time base counter	■ 64-bit up counter ■ 30-ns resolution
Interrupt controller		■ 3 timer interrupts ■ 7 incremental encoder index line interrupts ■ 1 UART (universal asynchronous receiver and transmitter) interrupt ■ 1 CAN interrupt ■ 1 slave DSP interrupt ■ 2 slave DSP PWM interrupts ■ 1 host interrupt ■ 4 external interrupts (user interrupts)
A/D converter	Channels	■ 16 multiplexed channels equipped with 4 sample & hold A/D converters (4 channels belong to one A/D converter. 4 consecutive samplings are necessary to sample all channels belonging to one A/D converter.) ■ 4 parallel channels each equipped with one sample & hold A/D converter ■ Note: 8 A/D converter channels (4 multiplexed and 4 parallel) can be sampled simultaneously.
	Resolution	■ 16-bit
	Input voltage range	■ ±10 V
	Overvoltage protection	■ ±15 V
	Conversion time	■ Multiplexed channels: 1 µs[1] ■ Parallel channels: 800 ns[1]
	Offset error	■ ±5 mV
	Gain error	■ ±0.25%
	Offset drift	■ 40 µV/K
	Gain drift	■ 50 ppm/K
	Signal-to-noise ratio	■ >83 dB
D/A converter	Channels	■ 8 channels
	Resolution	■ 16-bit
	Output range	■ ±10 V
	Settling time	■ 5 µs (14-bit)
	Offset error	■ ±1 mV
	Gain error	■ ±0.5%
	Offset drift	■ 30 µV/K
	Gain drift	■ 25 ppm/K

[1] *Speed and timing specifications describe the capabilities of the hardware components and circuits of our products. Depending on the software complexity, the attainable overall performance figures can deviate significantly from the hardware specifications.*



Parameter		Specification
D/A converter	Signal-to-noise ratio	■ >83 dB
	I_{max}	■ ±5 mA
	Cl_{max}	■ 10 nF
Digital I/O	Channels	■ 32-bit parallel I/O ■ Organized in four 8-bit groups ■ Each 8-bit group can be set to input or output (programmable by software)
	Voltage range	■ TTL input/output levels
	$I_{out\ max}$	■ ±10 mA
Digital incremental encoder interface	Channels	■ 6 independent channels ■ Single-ended (TTL) or differential (RS422) input (software programmable for each channel)
	Position counters	■ 24-bit resolution ■ Max. 1.65 MHz input frequency, i.e.,fourfold pulse count up to 6.6 MHz ■ Counter reset or reload via software
	Encoder supply voltage	■ 5 V/1.5 A ■ Shared with analog incremental encoder interface
Analog incremental encoder interface	Channels	■ 1 channel ■ Sinusoidal signals: 1 Vpp differential or 11 µApp differential (software programmable)
	Position counters	■ < 5° resolution ■ 32-bit loadable position counter ■ Max. 0.6 MHz input frequency, i.e., fourfold pulse count up to 2.4 MHz
	A/D converter performance	■ 6-bit resolution ■ 10 MSPS
	Encoder supply voltage	■ 5 V/1.5 A ■ Shared with digital incremental encoder interface
CAN interface	Configuration	■ 1 channel based on SAB 80C164 microcontroller ■ ISO DIS 11898-2 CAN high-speed standard
	Baud rate	■ Max. 1 Mbit/s
Serial interface	Configuration	■ TL6C550C single UART with FIFO ■ PLL-driven UART for accurate baud rate selection ■ RS232/RS422 compatibility
	Baud rate	■ Up to 115.2 kBd (RS232) ■ Up to 1 MBd (RS422)
Slave DSP	Type	■ Texas Instruments TMS320F240 DSP
	Clock rate	■ 20 MHz
	Memory	■ 64Kx16 external code memory ■ 28Kx16 external data memory ■ 4Kx16 dual-port memory for communication ■ 32 KB flash memory
	I/O channels	■ 16 A/D converter inputs ■ 10 PWM outputs ■ 4 capture inputs ■ 2 serial ports
	Input voltage range	■ TTL input/output level ■ A/D converter inputs: 0 … 5 V
	Output current	■ Max. ±13 mA
Host interface		■ Plug & Play support ■ Requires a full-size 16-bit ISA slot
Physical characteristics	Physical size	■ 340 x 125 x 45 mm (13.4 x 4.9 x 1.77 in)
	Ambient temperature	■ 0 … 50 °C (32 … 122 °F)
	Cooling	■ Passive cooling
	Power supply	■ +5 V ±5%, 4 A ■ +12 V ±5%, 0.75A ■ -12 V ±5%, 0.25A

Order Information

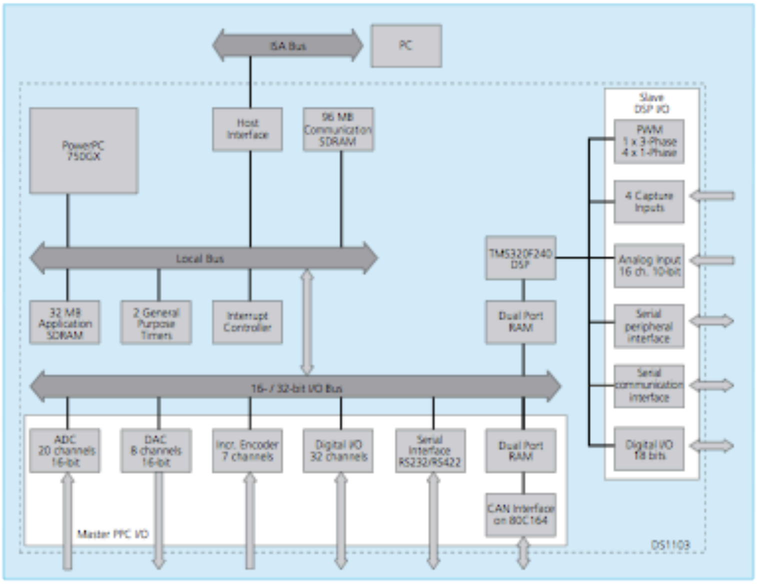
Product	Order Number
DS1103 PPC Controller Board	▪ DS1103

Relevant Software and Hardware

Software		Order Number
Included	▪ DS1103 Real-Time Library	–
	▪ Experiment and Platform Manager for hardware management	–
Required	▪ Real-Time Interface (RTI) (p. 156)	▪ RTI
	▪ Microtec C Compiler for PowerPC (p. 185)	▪ CCPPC
Optional	▪ Real-Time Interface CAN Blockset (p. 166)	▪ RTICAN_BS
	▪ Real-Time Interface CAN MultiMessage Blockset (p. 168)	▪ RTICANMM_BS
	▪ ControlDesk Standard – Developer Version (p. 186)	▪ CS_D
	▪ ControlDesk Standard – Operator Version (p. 186)	▪ CS_O
	▪ MLIB/MTRACE (p. 234)	▪ MLIB/MTRACE
	▪ CLIB (p. 233)	▪ CLIB
	▪ MotionDesk (p. 220)	▪ MotionDesk

Hardware		Order Number
Optional	▪ Connector Panel (p. 338)	▪ CP1103
	▪ Connector/LED Combi Panel (p. 338)	▪ CLP1103
	▪ Set of adapter cables for DS1103	▪ ADP_CAB1103

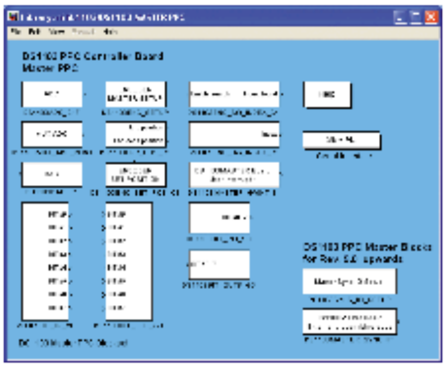
Block Diagram



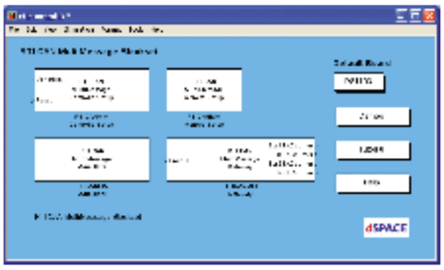
Graphical Configuration of the Controller Board

Using RTI

With Real-Time Interface (RTI) (p. 156), you can easily run your Simulink® models on the controller board. You can configure all I/O graphically by using RTI. Thereby, the implementation time is reduced to a minimum. With the RTI CAN Blockset (p. 166), CAN configurations can be completely carried out in a Simulink block diagram, with very little effort.


DS1103 PPC Controller Board
Master PPC
DS1103 PPC Master Blocks for Rev. 5.0 upwards

Real-Time Interface (p. 156) provides Simulink blocks for convenient configuration of items such as A/D, D/A, digital I/O lines, incremental encoder interface and PWM generation.



RTI CAN MultiMessage Blockset (p. 168) for graphical configuration of CAN interfaces.

Robotics

Rapid Prototyping in Robotics

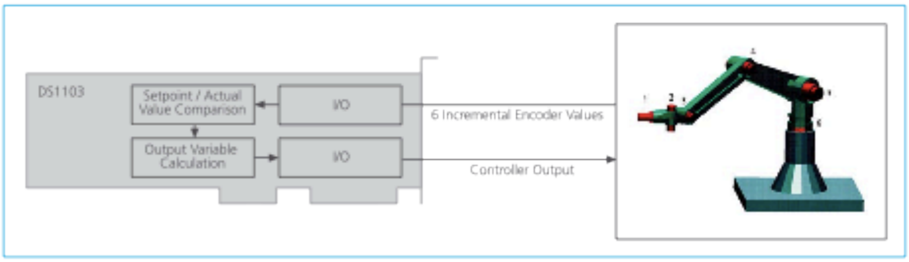
The DS1103 provides six digital incremental encoder interfaces. This is sufficient to pick up all the movements of a six-joint robot. Thus, this cost-effective single-board hardware makes it possible to perform rapid control prototyping in robotics.

Easy Handling

In the example below, the controller board replaces the position controller. The easy programmability of the DS1103 enables you to implement and test different control algorithms very quickly, which reduces design iteration times to a minimum. The prototyping hardware allows easy parameter changing and modification, without any hardware setup changes.

Calculating Values

The real-time system picks up the robot's six incremental encoder signals to determine the current robot position. Then this data is compared with the reference values. Afterwards, the DS1103 calculates the control algorithm and sends the controller output – for example, data on positions and velocities – back to the robot.

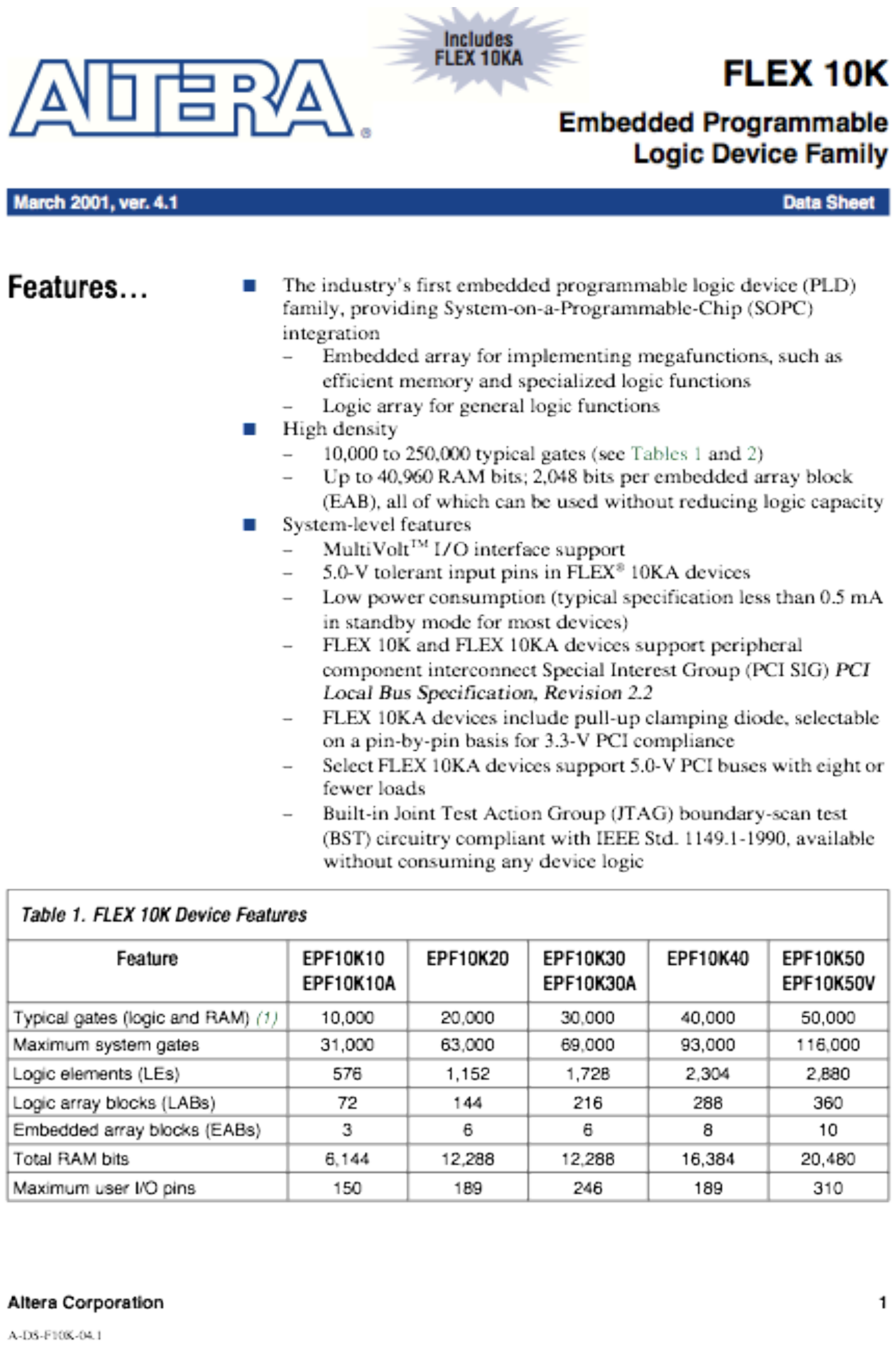

DS1103
Setpoint / Actual Value Comparison
I/O
Output Variable Calculation
I/O
6 Incremental Encoder Values
Controller Output

Calculating a control algorithm for robotics on a DS1103 PPC Controller Board.

Further Processing Potential

All reference values are calculated in real-time, even for inverse kinematics with highly nonlinear functions. External sensors such as axis-force momentum sensors can be included. Performing trajectory planning and advanced algorithms for collision avoidance is also very convenient with the DS1103 PPC Controller Board.

17.3. Altera EPF10K70 FPGA



Includes FLEX 10KA

ALTERA®

FLEX 10K

Embedded Programmable Logic Device Family

March 2001, ver. 4.1 **Data Sheet**

Features...

- The industry's first embedded programmable logic device (PLD) family, providing System-on-a-Programmable-Chip (SOPC) integration
  - Embedded array for implementing megafunctions, such as efficient memory and specialized logic functions
  - Logic array for general logic functions
- High density
  - 10,000 to 250,000 typical gates (see Tables 1 and 2)
  - Up to 40,960 RAM bits; 2,048 bits per embedded array block (EAB), all of which can be used without reducing logic capacity
- System-level features
  - MultiVolt™ I/O interface support
  - 5.0-V tolerant input pins in FLEX® 10KA devices
  - Low power consumption (typical specification less than 0.5 mA in standby mode for most devices)
  - FLEX 10K and FLEX 10KA devices support peripheral component interconnect Special Interest Group (PCI SIG) *PCI Local Bus Specification, Revision 2.2*
  - FLEX 10KA devices include pull-up clamping diode, selectable on a pin-by-pin basis for 3.3-V PCI compliance
  - Select FLEX 10KA devices support 5.0-V PCI buses with eight or fewer loads
  - Built-in Joint Test Action Group (JTAG) boundary-scan test (BST) circuitry compliant with IEEE Std. 1149.1-1990, available without consuming any device logic

Table 1. FLEX 10K Device Features

Feature	EPF10K10 EPF10K10A	EPF10K20	EPF10K30 EPF10K30A	EPF10K40	EPF10K50 EPF10K50V
Typical gates (logic and RAM) *(1)*	10,000	20,000	30,000	40,000	50,000
Maximum system gates	31,000	63,000	69,000	93,000	116,000
Logic elements (LEs)	576	1,152	1,728	2,304	2,880
Logic array blocks (LABs)	72	144	216	288	360
Embedded array blocks (EABs)	3	6	6	8	10
Total RAM bits	6,144	12,288	12,288	16,384	20,480
Maximum user I/O pins	150	189	246	189	310

Altera Corporation 1

A-DS-F10K-04.1

Table 2. FLEX 10K Device Features

Feature	EPF10K70	EPF10K100 EPF10K100A	EPF10K130V	EPF10K250A
Typical gates (logic and RAM) (1)	70,000	100,000	130,000	250,000
Maximum system gates	118,000	158,000	211,000	310,000
LEs	3,744	4,992	6,656	12,160
LABs	468	624	832	1,520
EABs	9	12	16	20
Total RAM bits	18,432	24,576	32,768	40,960
Maximum user I/O pins	358	406	470	470

Note to tables:

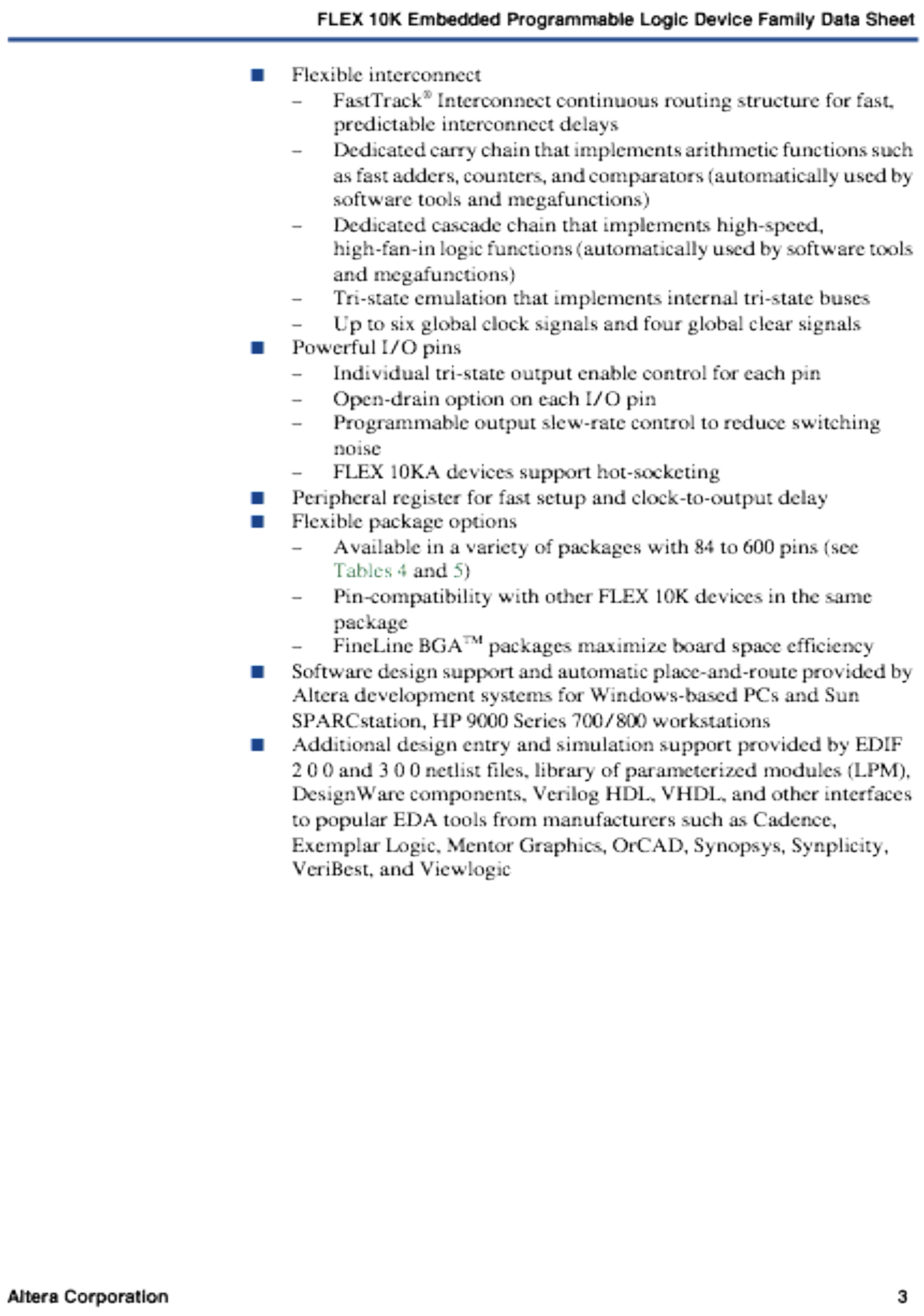
(1) The embedded IEEE Std. 1149.1 JTAG circuitry adds up to 31,250 gates in addition to the listed typical or maximum system gates.

...and More Features

- Devices are fabricated on advanced processes and operate with a 3.3-V or 5.0-V supply voltage (see Table 3
- In-circuit reconfigurability (ICR) via external configuration device, intelligent controller, or JTAG port
- ClockLock™ and ClockBoost™ options for reduced clock delay/skew and clock multiplication
- Built-in low-skew clock distribution trees
- 100% functional testing of all devices; test vectors or scan chains are not required

Table 3. Supply Voltages for FLEX 10K & FLEX 10KA Devices

5.0-V Devices	3.3-V Devices
EPF10K10	EPF10K10A
EPF10K20	EPF10K30A
EPF10K30	EPF10K50V
EPF10K40	EPF10K100A
EPF10K50	EPF10K130V
EPF10K70	EPF10K250A
EPF10K100	



- Flexible interconnect
  - FastTrack® Interconnect continuous routing structure for fast, predictable interconnect delays
  - Dedicated carry chain that implements arithmetic functions such as fast adders, counters, and comparators (automatically used by software tools and megafunctions)
  - Dedicated cascade chain that implements high-speed, high-fan-in logic functions (automatically used by software tools and megafunctions)
  - Tri-state emulation that implements internal tri-state buses
  - Up to six global clock signals and four global clear signals
- Powerful I/O pins
  - Individual tri-state output enable control for each pin
  - Open-drain option on each I/O pin
  - Programmable output slew-rate control to reduce switching noise
  - FLEX 10KA devices support hot-socketing
- Peripheral register for fast setup and clock-to-output delay
- Flexible package options
  - Available in a variety of packages with 84 to 600 pins (see Tables 4 and 5)
  - Pin-compatibility with other FLEX 10K devices in the same package
  - FineLine BGA™ packages maximize board space efficiency
- Software design support and automatic place-and-route provided by Altera development systems for Windows-based PCs and Sun SPARCstation, HP 9000 Series 700/800 workstations
- Additional design entry and simulation support provided by EDIF 2 0 0 and 3 0 0 netlist files, library of parameterized modules (LPM), DesignWare components, Verilog HDL, VHDL, and other interfaces to popular EDA tools from manufacturers such as Cadence, Exemplar Logic, Mentor Graphics, OrCAD, Synopsys, Synplicity, VeriBest, and Viewlogic

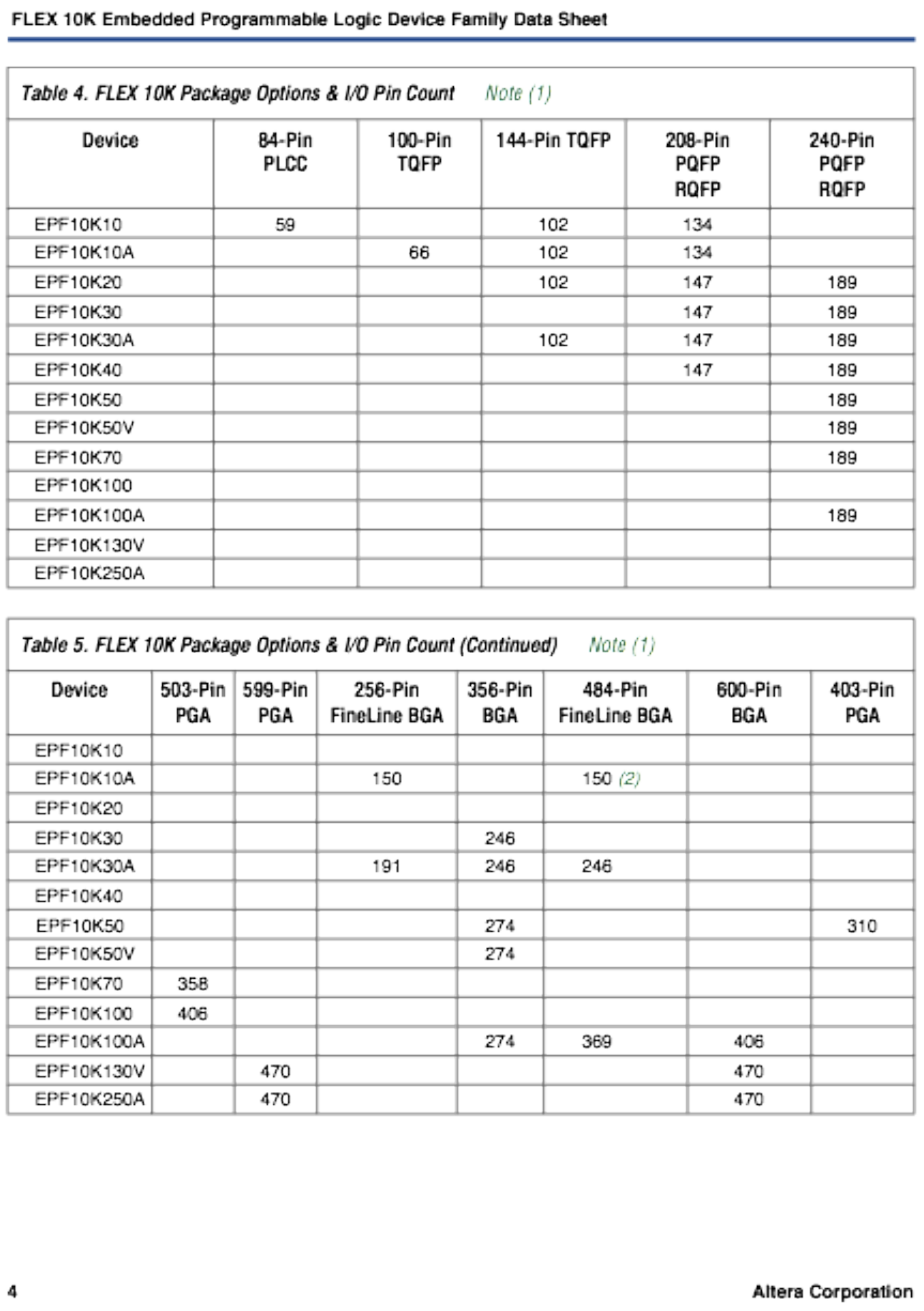
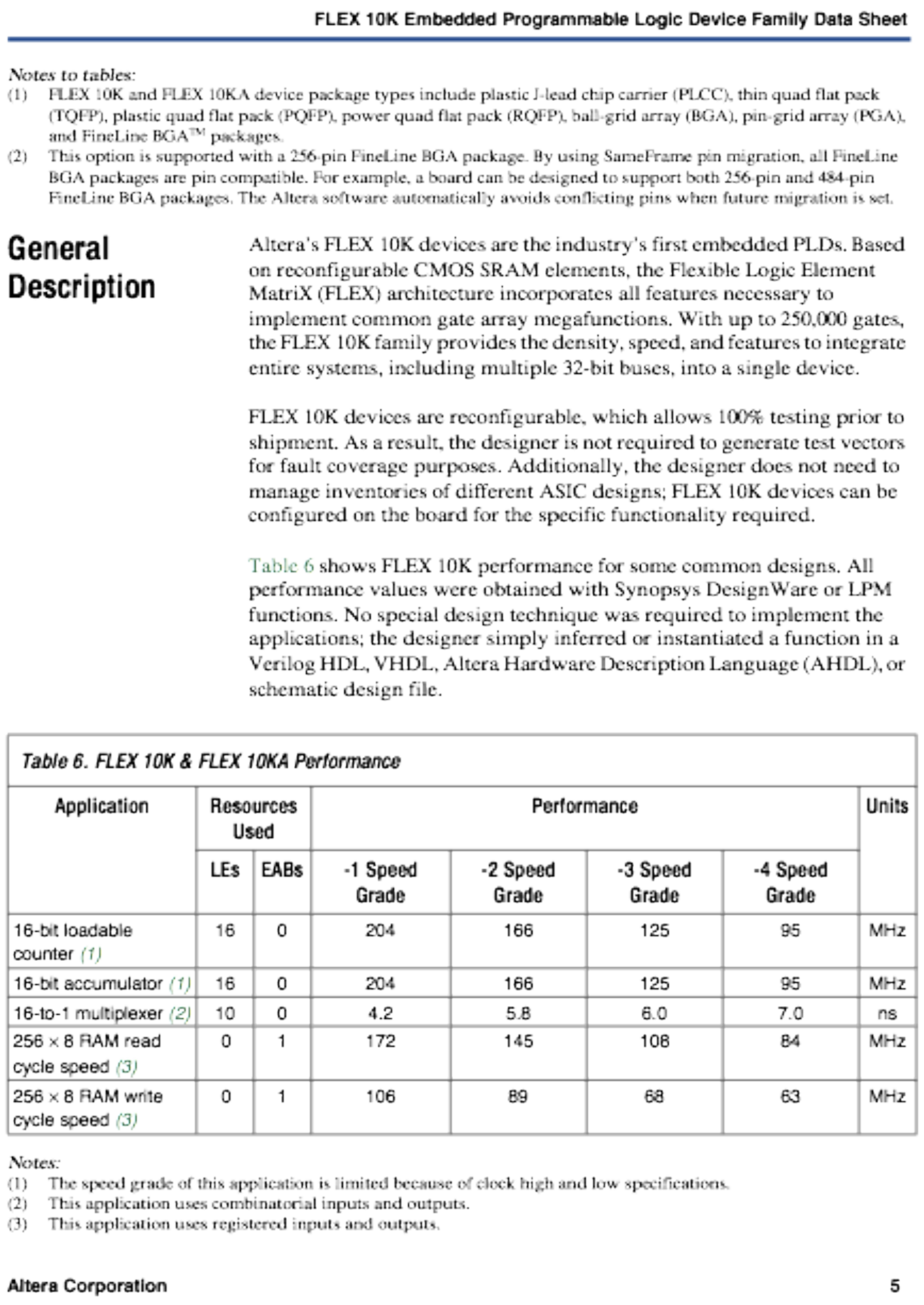


Table 4. FLEX 10K Package Options & I/O Pin Count Note (1)

Device	84-Pin PLCC	100-Pin TQFP	144-Pin TQFP	208-Pin PQFP RQFP	240-Pin PQFP RQFP
EPF10K10	59		102	134	
EPF10K10A		66	102	134	
EPF10K20			102	147	189
EPF10K30				147	189
EPF10K30A			102	147	189
EPF10K40				147	189
EPF10K50					189
EPF10K50V					189
EPF10K70					189
EPF10K100					
EPF10K100A					189
EPF10K130V					
EPF10K250A					

Table 5. FLEX 10K Package Options & I/O Pin Count (Continued) Note (1)

Device	503-Pin PGA	599-Pin PGA	256-Pin FineLine BGA	356-Pin BGA	484-Pin FineLine BGA	600-Pin BGA	403-Pin PGA
EPF10K10							
EPF10K10A			150		150 (2)		
EPF10K20							
EPF10K30				246			
EPF10K30A			191	246	246		
EPF10K40							
EPF10K50				274			310
EPF10K50V				274			
EPF10K70	358						
EPF10K100	406						
EPF10K100A				274	369	406	
EPF10K130V		470				470	
EPF10K250A		470				470	



Notes to tables:
(1) FLEX 10K and FLEX 10KA device package types include plastic J-lead chip carrier (PLCC), thin quad flat pack (TQFP), plastic quad flat pack (PQFP), power quad flat pack (RQFP), ball-grid array (BGA), pin-grid array (PGA), and FineLine BGA™ packages.
(2) This option is supported with a 256-pin FineLine BGA package. By using SameFrame pin migration, all FineLine BGA packages are pin compatible. For example, a board can be designed to support both 256-pin and 484-pin FineLine BGA packages. The Altera software automatically avoids conflicting pins when future migration is set.

General Description

Altera's FLEX 10K devices are the industry's first embedded PLDs. Based on reconfigurable CMOS SRAM elements, the Flexible Logic Element MatriX (FLEX) architecture incorporates all features necessary to implement common gate array megafunctions. With up to 250,000 gates, the FLEX 10K family provides the density, speed, and features to integrate entire systems, including multiple 32-bit buses, into a single device.

FLEX 10K devices are reconfigurable, which allows 100% testing prior to shipment. As a result, the designer is not required to generate test vectors for fault coverage purposes. Additionally, the designer does not need to manage inventories of different ASIC designs; FLEX 10K devices can be configured on the board for the specific functionality required.

Table 6 shows FLEX 10K performance for some common designs. All performance values were obtained with Synopsys DesignWare or LPM functions. No special design technique was required to implement the applications; the designer simply inferred or instantiated a function in a Verilog HDL, VHDL, Altera Hardware Description Language (AHDL), or schematic design file.

Table 6. FLEX 10K & FLEX 10KA Performance

Application	Resources Used		Performance				Units
	LEs	EABs	-1 Speed Grade	-2 Speed Grade	-3 Speed Grade	-4 Speed Grade	
16-bit loadable counter (1)	16	0	204	166	125	95	MHz
16-bit accumulator (1)	16	0	204	166	125	95	MHz
16-to-1 multiplexer (2)	10	0	4.2	5.8	6.0	7.0	ns
256 × 8 RAM read cycle speed (3)	0	1	172	145	108	84	MHz
256 × 8 RAM write cycle speed (3)	0	1	106	89	68	63	MHz

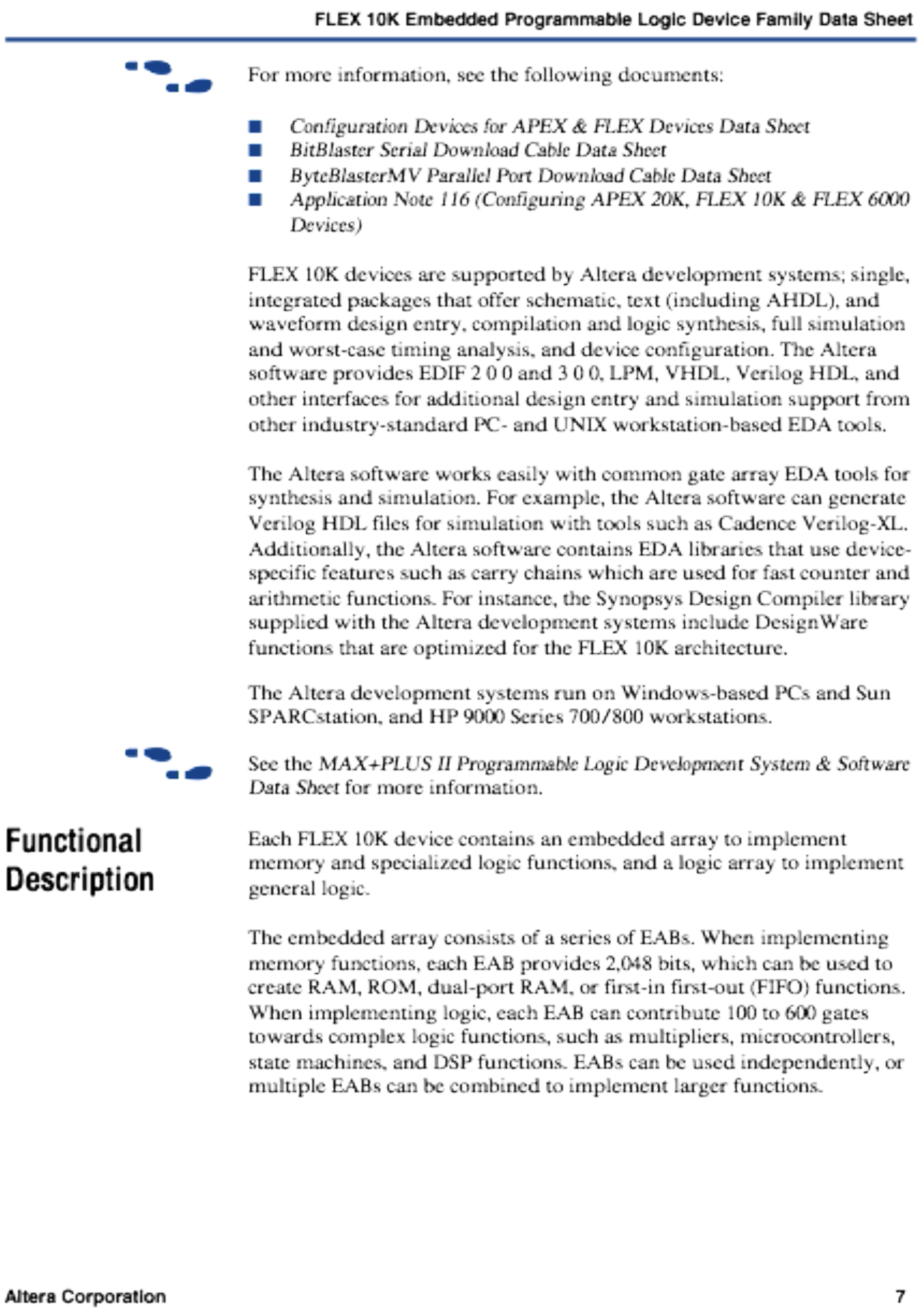
Notes:
(1) The speed grade of this application is limited because of clock high and low specifications.
(2) This application uses combinatorial inputs and outputs.
(3) This application uses registered inputs and outputs.

The FLEX 10K architecture is similar to that of embedded gate arrays, the fastest-growing segment of the gate array market. As with standard gate arrays, embedded gate arrays implement general logic in a conventional "sea-of-gates" architecture. In addition, embedded gate arrays have dedicated die areas for implementing large, specialized functions. By embedding functions in silicon, embedded gate arrays provide reduced die area and increased speed compared to standard gate arrays. However, embedded megafunctions typically cannot be customized, limiting the designer's options. In contrast, FLEX 10K devices are programmable, providing the designer with full control over embedded megafunctions and general logic while facilitating iterative design changes during debugging.

Each FLEX 10K device contains an embedded array and a logic array. The embedded array is used to implement a variety of memory functions or complex logic functions, such as digital signal processing (DSP), microcontroller, wide-data-path manipulation, and data-transformation functions. The logic array performs the same function as the sea-of-gates in the gate array: it is used to implement general logic, such as counters, adders, state machines, and multiplexers. The combination of embedded and logic arrays provides the high performance and high density of embedded gate arrays, enabling designers to implement an entire system on a single device.

FLEX 10K devices are configured at system power-up with data stored in an Altera serial configuration device or provided by a system controller. Altera offers the EPC1, EPC2, EPC16, and EPC1441 configuration devices, which configure FLEX 10K devices via a serial data stream. Configuration data can also be downloaded from system RAM or from Altera's BitBlaster™ serial download cable or ByteBlasterMV™ parallel port download cable. After a FLEX 10K device has been configured, it can be reconfigured in-circuit by resetting the device and loading new data. Because reconfiguration requires less than 320 ms, real-time changes can be made during system operation.

FLEX 10K devices contain an optimized interface that permits microprocessors to configure FLEX 10K devices serially or in parallel, and synchronously or asynchronously. The interface also enables microprocessors to treat a FLEX 10K device as memory and configure the device by writing to a virtual memory location, making it very easy for the designer to reconfigure the device.



For more information, see the following documents:

- *Configuration Devices for APEX & FLEX Devices Data Sheet*
- *BitBlaster Serial Download Cable Data Sheet*
- *ByteBlasterMV Parallel Port Download Cable Data Sheet*
- *Application Note 116 (Configuring APEX 20K, FLEX 10K & FLEX 6000 Devices)*

FLEX 10K devices are supported by Altera development systems; single, integrated packages that offer schematic, text (including AHDL), and waveform design entry, compilation and logic synthesis, full simulation and worst-case timing analysis, and device configuration. The Altera software provides EDIF 2 0 0 and 3 0 0, LPM, VHDL, Verilog HDL, and other interfaces for additional design entry and simulation support from other industry-standard PC- and UNIX workstation-based EDA tools.

The Altera software works easily with common gate array EDA tools for synthesis and simulation. For example, the Altera software can generate Verilog HDL files for simulation with tools such as Cadence Verilog-XL. Additionally, the Altera software contains EDA libraries that use device-specific features such as carry chains which are used for fast counter and arithmetic functions. For instance, the Synopsys Design Compiler library supplied with the Altera development systems include DesignWare functions that are optimized for the FLEX 10K architecture.

The Altera development systems run on Windows-based PCs and Sun SPARCstation, and HP 9000 Series 700/800 workstations.

See the *MAX+PLUS II Programmable Logic Development System & Software Data Sheet* for more information.

Functional Description

Each FLEX 10K device contains an embedded array to implement memory and specialized logic functions, and a logic array to implement general logic.

The embedded array consists of a series of EABs. When implementing memory functions, each EAB provides 2,048 bits, which can be used to create RAM, ROM, dual-port RAM, or first-in first-out (FIFO) functions. When implementing logic, each EAB can contribute 100 to 600 gates towards complex logic functions, such as multipliers, microcontrollers, state machines, and DSP functions. EABs can be used independently, or multiple EABs can be combined to implement larger functions.

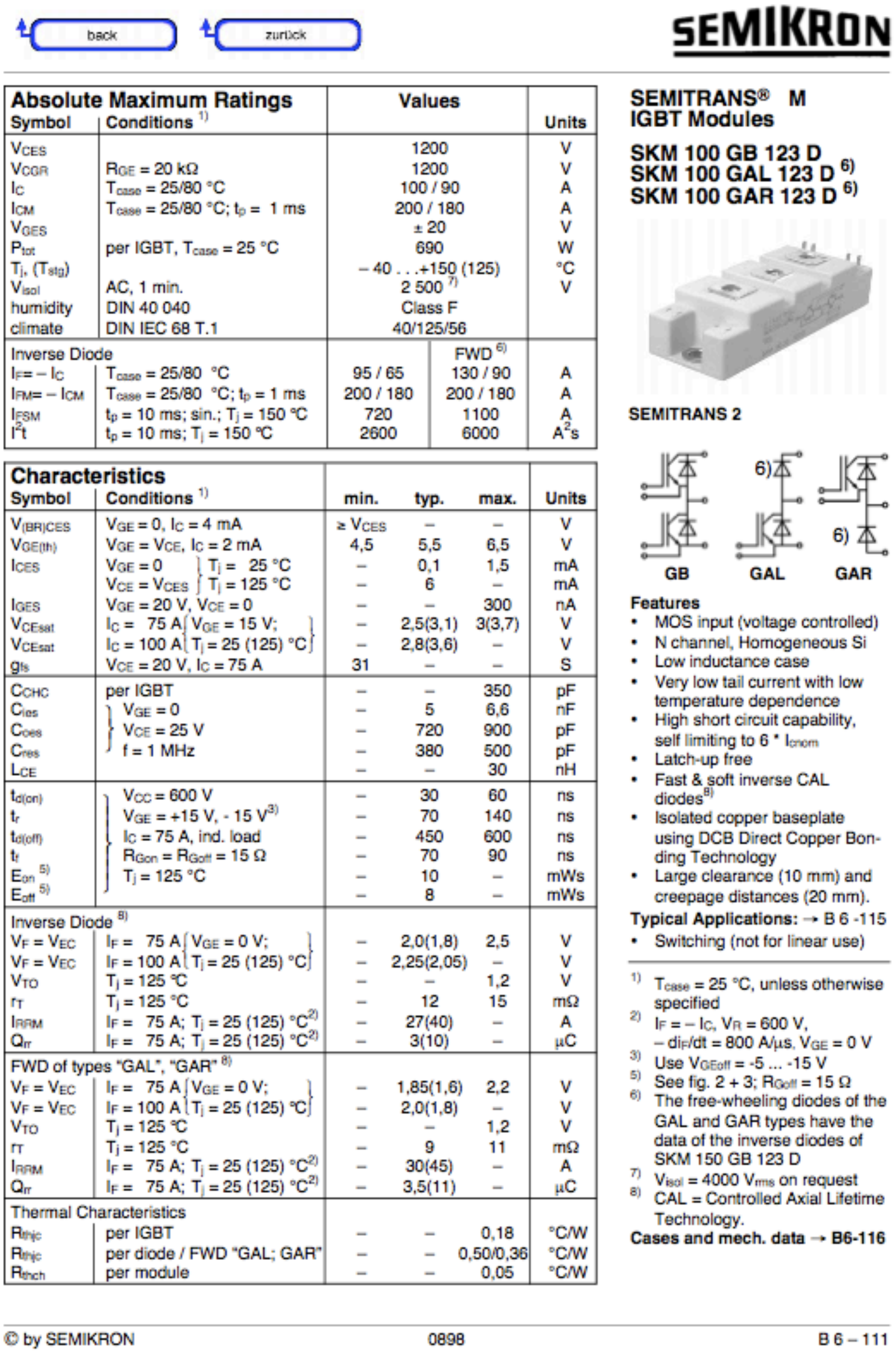
The logic array consists of logic array blocks (LABs). Each LAB contains eight LEs and a local interconnect. An LE consists of a 4-input look-up table (LUT), a programmable flipflop, and dedicated signal paths for carry and cascade functions. The eight LEs can be used to create medium-sized blocks of logic—8-bit counters, address decoders, or state machines—or combined across LABs to create larger logic blocks. Each LAB represents about 96 usable gates of logic.

Signal interconnections within FLEX 10K devices and to and from device pins are provided by the FastTrack Interconnect, a series of fast, continuous row and column channels that run the entire length and width of the device.

Each I/O pin is fed by an I/O element (IOE) located at the end of each row and column of the FastTrack Interconnect. Each IOE contains a bidirectional I/O buffer and a flipflop that can be used as either an output or input register to feed input, output, or bidirectional signals. When used with a dedicated clock pin, these registers provide exceptional performance. As inputs, they provide setup times as low as 1.6 ns and hold times of 0 ns; as outputs, these registers provide clock-to-output times as low as 5.3 ns. IOEs provide a variety of features, such as JTAG BST support, slew-rate control, tri-state buffers, and open-drain outputs.

Figure 1 shows a block diagram of the FLEX 10K architecture. Each group of LEs is combined into an LAB; LABs are arranged into rows and columns. Each row also contains a single EAB. The LABs and EABs are interconnected by the FastTrack Interconnect. IOEs are located at the end of each row and column of the FastTrack Interconnect.

17.4. Semikron SKM 100 IGBT



back zurück

SEMIKRON

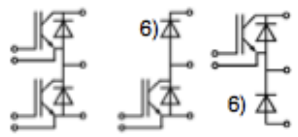
Absolute Maximum Ratings Symbol	Conditions [1]	Values		Units
V_{CES}		1200		V
V_{CGR}	R_{GE} = 20 kΩ	1200		V
I_C	T_{case} = 25/80 °C	100 / 90		A
I_{CM}	T_{case} = 25/80 °C; t_p = 1 ms	200 / 180		A
V_{GES}		± 20		V
P_{tot}	per IGBT, T_{case} = 25 °C	690		W
T_j, (T_{stg})		– 40 . . .+150 (125)		°C
V_{isol}	AC, 1 min.	2 500 [7]		V
humidity	DIN 40 040	Class F		
climate	DIN IEC 68 T.1	40/125/56		
Inverse Diode			FWD [6]	
$I_F = -I_C$	T_{case} = 25/80 °C	95 / 65	130 / 90	A
$I_{FM} = -I_{CM}$	T_{case} = 25/80 °C; t_p = 1 ms	200 / 180	200 / 180	A
I_{FSM}	t_p = 10 ms; sin.; T_j = 150 °C	720	1100	A
I^2t	t_p = 10 ms; T_j = 150 °C	2600	6000	A^2s

Characteristics Symbol	Conditions [1]	min.	typ.	max.	Units
$V_{(BR)CES}$	V_{GE} = 0, I_C = 4 mA	≥ V_{CES}	–	–	V
$V_{GE(th)}$	$V_{GE} = V_{CE}$, I_C = 2 mA	4,5	5,5	6,5	V
I_{CES}	V_{GE} = 0 ; T_j = 25 °C	–	0,1	1,5	mA
	$V_{CE} = V_{CES}$; T_j = 125 °C	–	6	–	mA
I_{GES}	V_{GE} = 20 V, V_{CE} = 0	–	–	300	nA
V_{CEsat}	I_C = 75 A ; V_{GE} = 15 V;	–	2,5(3,1)	3(3,7)	V
V_{CEsat}	I_C = 100 A ; T_j = 25 (125) °C	–	2,8(3,6)	–	V
g_{fs}	V_{CE} = 20 V, I_C = 75 A	31	–	–	S
C_{CHC}	per IGBT	–	–	350	pF
C_{ies}	V_{GE} = 0	–	5	6,6	nF
C_{oes}	V_{CE} = 25 V	–	720	900	pF
C_{res}	f = 1 MHz	–	380	500	pF
L_{CE}		–	–	30	nH
$t_{d(on)}$	V_{CC} = 600 V	–	30	60	ns
t_r	V_{GE} = +15 V, - 15 V [3]	–	70	140	ns
$t_{d(off)}$	I_C = 75 A, ind. load	–	450	600	ns
t_f	$R_{Gon} = R_{Goff}$ = 15 Ω	–	70	90	ns
E_{on} [5]	T_j = 125 °C	–	10	–	mWs
E_{off} [5]		–	8	–	mWs
Inverse Diode [8]					
$V_F = V_{EC}$	I_F = 75 A ; V_{GE} = 0 V;	–	2,0(1,8)	2,5	V
$V_F = V_{EC}$	I_F = 100 A ; T_j = 25 (125) °C	–	2,25(2,05)	–	V
V_{TO}	T_j = 125 °C	–	–	1,2	V
r_T	T_j = 125 °C	–	12	15	mΩ
I_{RRM}	I_F = 75 A; T_j = 25 (125) °C [2]	–	27(40)	–	A
Q_{rr}	I_F = 75 A; T_j = 25 (125) °C [2]	–	3(10)	–	µC
FWD of types "GAL", "GAR" [8]					
$V_F = V_{EC}$	I_F = 75 A ; V_{GE} = 0 V;	–	1,85(1,6)	2,2	V
$V_F = V_{EC}$	I_F = 100 A ; T_j = 25 (125) °C	–	2,0(1,8)	–	V
V_{TO}	T_j = 125 °C	–	–	1,2	V
r_T	T_j = 125 °C	–	9	11	mΩ
I_{RRM}	I_F = 75 A; T_j = 25 (125) °C [2]	–	30(45)	–	A
Q_{rr}	I_F = 75 A; T_j = 25 (125) °C [2]	–	3,5(11)	–	µC
Thermal Characteristics					
R_{thjc}	per IGBT	–	–	0,18	°C/W
R_{thjc}	per diode / FWD "GAL; GAR"	–	–	0,50/0,36	°C/W
R_{thch}	per module	–	–	0,05	°C/W

SEMITRANS® M
IGBT Modules

SKM 100 GB 123 D
SKM 100 GAL 123 D [6]
SKM 100 GAR 123 D [6]

SEMITRANS 2



GB GAL GAR

Features

- MOS input (voltage controlled)
- N channel, Homogeneous Si
- Low inductance case
- Very low tail current with low temperature dependence
- High short circuit capability, self limiting to 6 * I_{cnom}
- Latch-up free
- Fast & soft inverse CAL diodes [8]
- Isolated copper baseplate using DCB Direct Copper Bonding Technology
- Large clearance (10 mm) and creepage distances (20 mm).

Typical Applications: → B 6 -115

- Switching (not for linear use)

[1] T_{case} = 25 °C, unless otherwise specified
[2] $I_F = -I_C$, V_R = 600 V, $-di_F/dt$ = 800 A/µs, V_{GE} = 0 V
[3] Use V_{GEoff} = -5 ... -15 V
[5] See fig. 2 + 3; R_{Goff} = 15 Ω
[6] The free-wheeling diodes of the GAL and GAR types have the data of the inverse diodes of SKM 150 GB 123 D
[7] V_{isol} = 4000 V_{rms} on request
[8] CAL = Controlled Axial Lifetime Technology.

Cases and mech. data → B6-116

© by SEMIKRON 0898 B 6 – 111

SKM 100 GB 123 D…

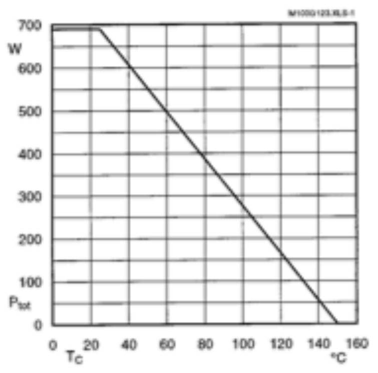
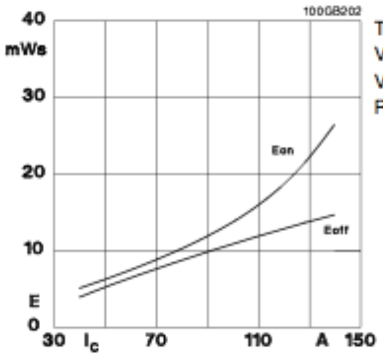
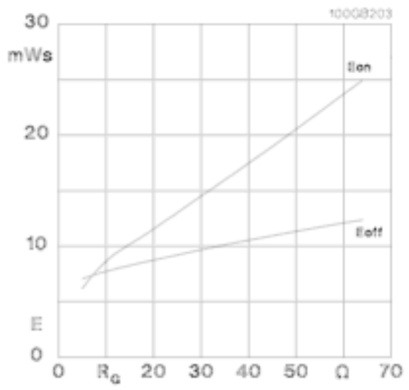


Fig. 1 Rated power dissipation $P_{tot} = f(T_C)$



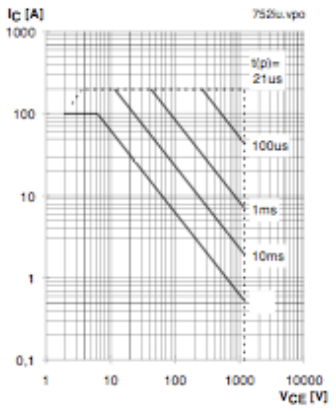
$T_j = 125$ °C
$V_{CE} = 600$ V
$V_{GE} = \pm 15$ V
$R_G = 15\ \Omega$

Fig. 2 Turn-on /-off energy = f (I_C)



$T_j = 125$ °C
$V_{CE} = 600$ V
$V_{GE} = \pm 15$ V
$I_C = 75$ A

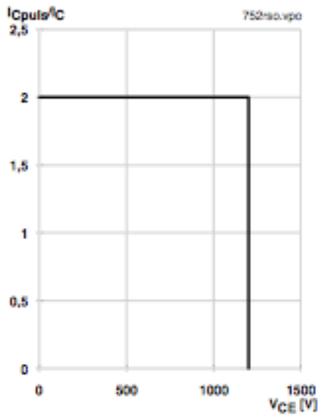
Fig. 3 Turn-on /-off energy = f (R_G)



1 pulse
$T_C = 25$ °C
$T_j \leq 150$ °C

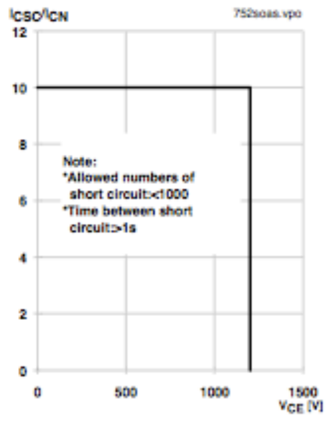
Not for linear use

Fig. 4 Maximum safe operating area (SOA) $I_C = f(V_{CE})$



$T_j \leq 150$ °C
$V_{GE} = 15$ V
$R_{Goff} = 15\ \Omega$
$I_C = 75$ A

Fig. 5 Turn-off safe operating area (RBSOA)



$T_j \leq 150$ °C
$V_{GE} = \pm 15$ V
$t_{sc} \leq 10$ µs
$L < 25$ nH
$I_{CN} = 75$ A

Fig. 6 Safe operating area at short circuit $I_C = f(V_{CE})$

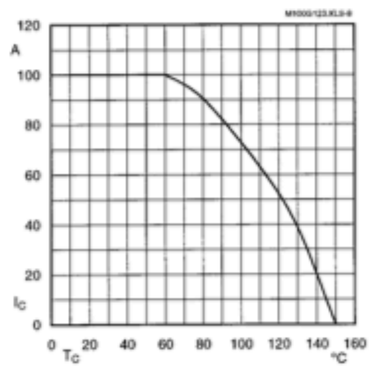


Fig. 8 Rated current vs. temperature $I_C = f(T_C)$

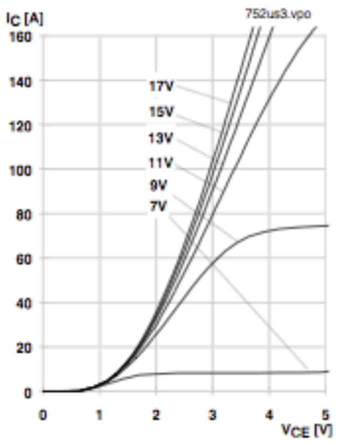


Fig. 9 Typ. output characteristic, t_p = 80 µs; 25 °C

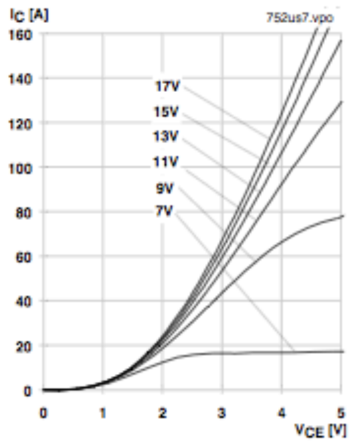


Fig. 10 Typ. output characteristic, t_p = 80 µs; 125 °C

$P_{cond(t)} = V_{CEsat(t)} \cdot I_{C(t)}$

$V_{CEsat(t)} = V_{CE(TO)(Tj)} + r_{CE(Tj)} \cdot I_{C(t)}$

$V_{CE(TO)(Tj)} \leq 1{,}5 + 0{,}002\ (T_j - 25)$ [V]

typ.: $r_{CE(Tj)} = 0{,}013 + 0{,}00005\ (T_j - 25)$ [Ω]

max.: $r_{CE(Tj)} = 0{,}020 + 0{,}00007\ (T_j - 25)$ [Ω]

valid for $V_{GE} = +15\,^{+2}_{-1}$ [V]; $I_C > 0{,}3\ I_{Cnom}$

Fig. 11 Saturation characteristic (IGBT)
Calculation elements and equations

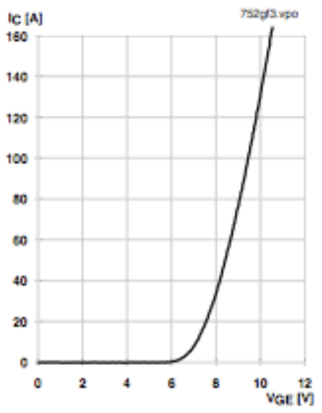
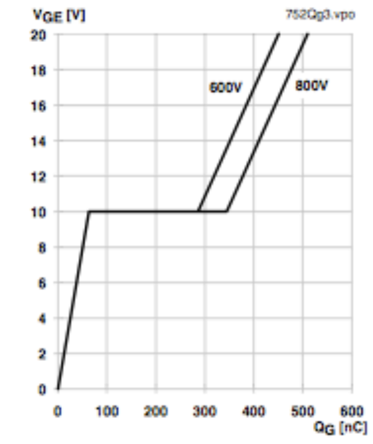


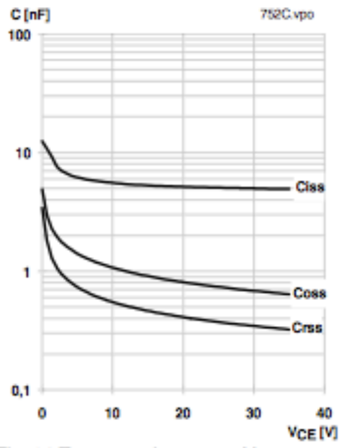
Fig. 12 Typ. transfer characteristic, t_p = 80 µs; V_{CE} = 20 V

SKM 100 GB 123 D…



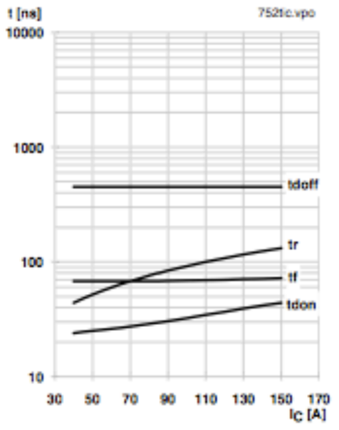
I_{Cpuls} = 75 A

Fig. 13 Typ. gate charge characteristic



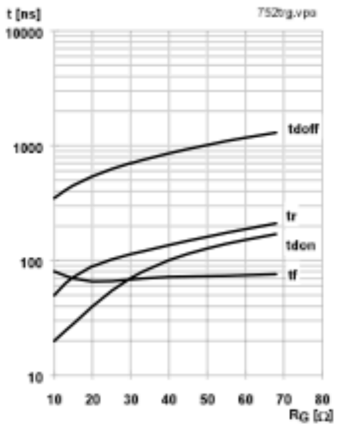
V_{GE} = 0 V
f = 1 MHZ

Fig. 14 Typ. capacitances vs.V_{CE}



T_j = 125 °C
V_{CE} = 600 V
V_{GE} = ± 15 V
R_{Gon} = 15 Ω
R_{Goff} = 15 Ω
induct. load

Fig. 15 Typ. switching times vs. I_C



T_j = 125 °C
V_{CE} = 600 V
V_{GE} = ± 15 V
I_C = 75 A
induct. load

Fig. 16 Typ. switching times vs. gate resistor R_G

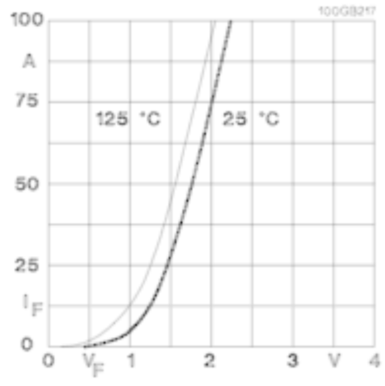


Fig. 17 Typ. CAL diode forward characteristic

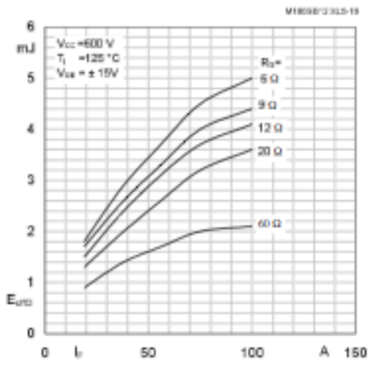


Fig. 18 Diode turn-off energy dissipation per pulse

SEMIKRON

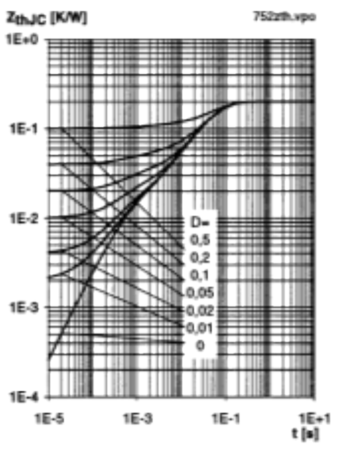


Fig. 19 Transient thermal impedance of IGBT $Z_{thJC} = f(t_p)$; $D = t_p / t_c = t_p \cdot f$

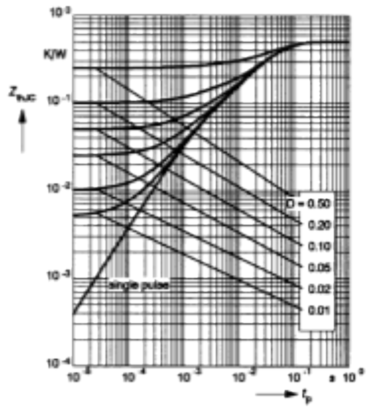


Fig. 20 Transient thermal impedance of inverse CAL diodes $Z_{thJC} = f(t_p)$; $D = t_p / t_c = t_p \cdot f$

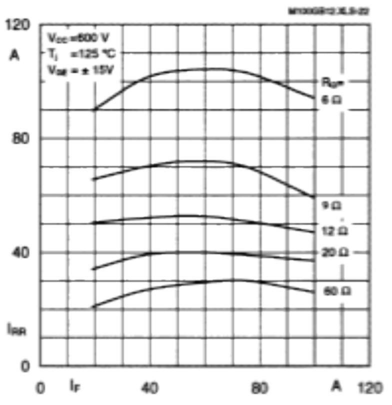


Fig. 22 Typ. CAL diode peak reverse recovery current $I_{RR} = f(I_F; R_G)$

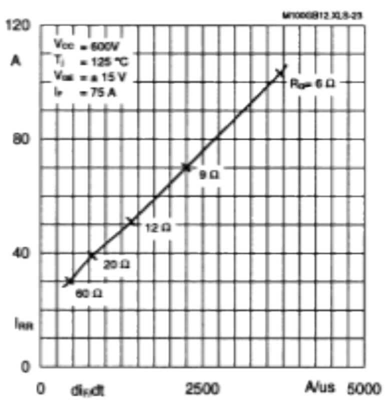


Fig. 23 Typ. CAL diode peak reverse recovery current $I_{RR} = f(di_F/dt)$

Typical Applications include

Switched mode power supplies
DC servo and robot drives
Inverters
DC choppers (versions GAR; GAL)
AC motor speed control
Inductive heating
UPS Uninterruptable power supplies
General power switching applications
Electronic (also portable) welders
Pulse frequencies also above 15 kHz

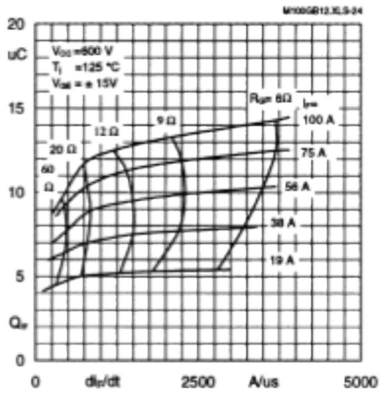
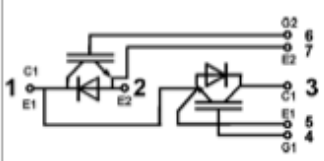


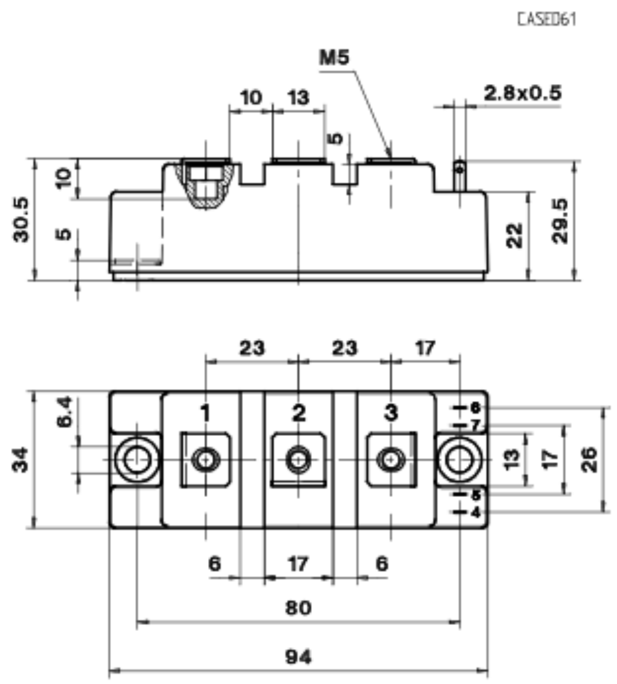
Fig. 24 Typ. CAL diode recovered charge $Q_{rr} = f(di/dt)$

SKM 100 GB 123 D…

SEMITRANS 2
Case D 61
UL Recognized
File no. E 63 532

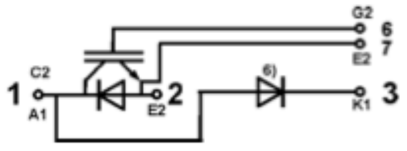
SKM 100 GB 123 D
SKM 100 GB 173 D



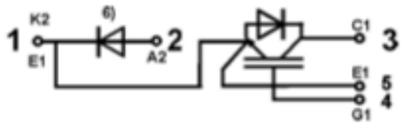


Dimensions in mm

SKM 100 GAL 123 D
Case D 62 (→ D 61)



SKM 100 GAR 123 D
Case D 63 (→ D 61)



Case outline and circuit diagrams

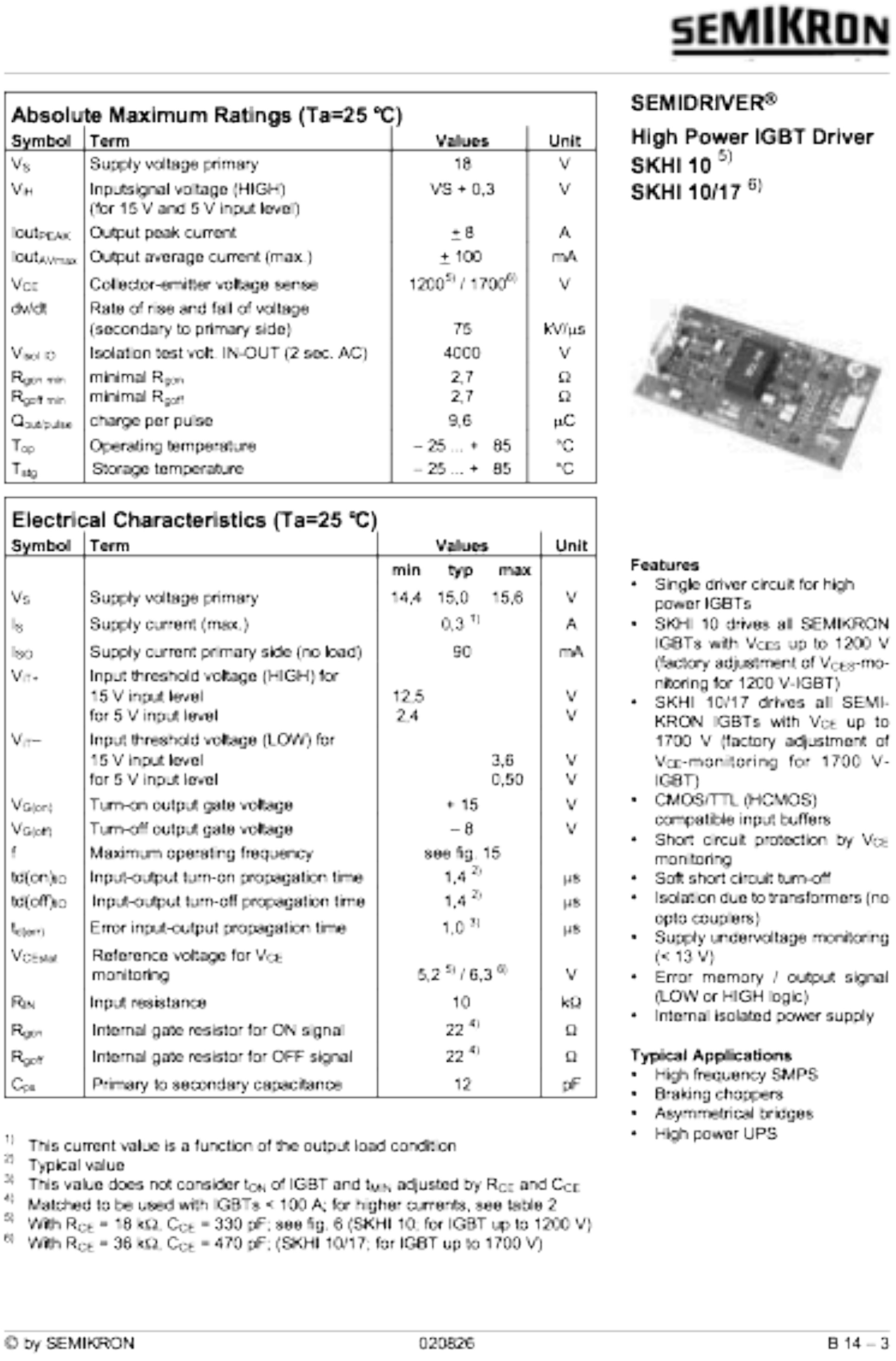
Mechanical Data					
Symbol	**Conditions**	**Values** min.	typ.	max.	**Units**
M_1	to heatsink, SI Units (M6)	3	–	5	Nm
	to heatsink, US Units	27	–	44	lb.in.
M_2	for terminals, SI Units (M5)	2,5	–	5	Nm
	for terminals US Units	22	–	44	lb.in.
a		–	–	5x9,81	m/s^2
w		–	–	160	g

[6)] Freewheeling diode → B 6 - 111, remark 6.

This is an electrostatic discharge sensitive device (ESDS). Please observe the international standard IEC 747-1, Chapter IX.

Eight devices are supplied in one SEMIBOX A without mounting hardware, which can be ordered separately under Ident No. 33321100 (for 10 SEMITRANS 2). Larger packing units of 20 and 42 pieces are used if suitable
Accessories → B 6 - 4.
SEMIBOX → C - 1.

17.5. Semikron SKHI 10 Drivers



SEMIKRON

SEMIDRIVER®

High Power IGBT Driver

SKHI 10 [5)]

SKHI 10/17 [6)]

Absolute Maximum Ratings (Ta=25 °C)

Symbol	Term	Values	Unit
V_S	Supply voltage primary	18	V
V_{iH}	Inputsignal voltage (HIGH) (for 15 V and 5 V input level)	VS + 0,3	V
$Iout_{PEAK}$	Output peak current	± 8	A
$Iout_{AVmax}$	Output average current (max.)	± 100	mA
V_{CE}	Collector-emitter voltage sense	1200[5)] / 1700[6)]	V
dv/dt	Rate of rise and fall of voltage (secondary to primary side)	75	kV/µs
$V_{isol\ IO}$	Isolation test volt. IN-OUT (2 sec. AC)	4000	V
$R_{gon\ min}$	minimal R_{gon}	2,7	Ω
$R_{goff\ min}$	minimal R_{goff}	2,7	Ω
$Q_{out/pulse}$	charge per pulse	9,6	µC
T_{op}	Operating temperature	– 25 ... + 85	°C
T_{stg}	Storage temperature	– 25 ... + 85	°C

Electrical Characteristics (Ta=25 °C)

Symbol	Term	Values			Unit
		min	typ	max	
V_S	Supply voltage primary	14,4	15,0	15,6	V
I_S	Supply current (max.)		0,3 [1)]		A
I_{SO}	Supply current primary side (no load)		90		mA
V_{iT+}	Input threshold voltage (HIGH) for				
	15 V input level	12,5			V
	for 5 V input level	2,4			V
V_{iT-}	Input threshold voltage (LOW) for				
	15 V input level			3,6	V
	for 5 V input level			0,50	V
$V_{G(on)}$	Turn-on output gate voltage		+ 15		V
$V_{G(off)}$	Turn-off output gate voltage		– 8		V
f	Maximum operating frequency		see fig. 15		
$td(on)_{IO}$	Input-output turn-on propagation time		1,4 [2)]		µs
$td(off)_{IO}$	Input-output turn-off propagation time		1,4 [2)]		µs
$t_{d(err)}$	Error input-output propagation time		1,0 [3)]		µs
V_{CEstat}	Reference voltage for V_{CE} monitoring		5,2 [5)] / 6,3 [6)]		V
R_{IN}	Input resistance		10		kΩ
R_{gon}	Internal gate resistor for ON signal		22 [4)]		Ω
R_{goff}	Internal gate resistor for OFF signal		22 [4)]		Ω
C_{ps}	Primary to secondary capacitance		12		pF

1) This current value is a function of the output load condition
2) Typical value
3) This value does not consider t_{ON} of IGBT and t_{MIN} adjusted by R_{CE} and C_{CE}
4) Matched to be used with IGBTs < 100 A; for higher currents, see table 2
5) With R_{CE} = 18 kΩ, C_{CE} = 330 pF; see fig. 6 (SKHI 10; for IGBT up to 1200 V)
6) With R_{CE} = 36 kΩ, C_{CE} = 470 pF; (SKHI 10/17; for IGBT up to 1700 V)

Features
- Single driver circuit for high power IGBTs
- SKHI 10 drives all SEMIKRON IGBTs with V_{CES} up to 1200 V (factory adjustment of V_{CES}-monitoring for 1200 V-IGBT)
- SKHI 10/17 drives all SEMIKRON IGBTs with V_{CE} up to 1700 V (factory adjustment of V_{CE}-monitoring for 1700 V-IGBT)
- CMOS/TTL (HCMOS) compatible input buffers
- Short circuit protection by V_{CE} monitoring
- Soft short circuit turn-off
- Isolation due to transformers (no opto couplers)
- Supply undervoltage monitoring (< 13 V)
- Error memory / output signal (LOW or HIGH logic)
- Internal isolated power supply

Typical Applications
- High frequency SMPS
- Braking choppers
- Asymmetrical bridges
- High power UPS

© by SEMIKRON 020826 B 14 – 3

Block diagram SKHI10

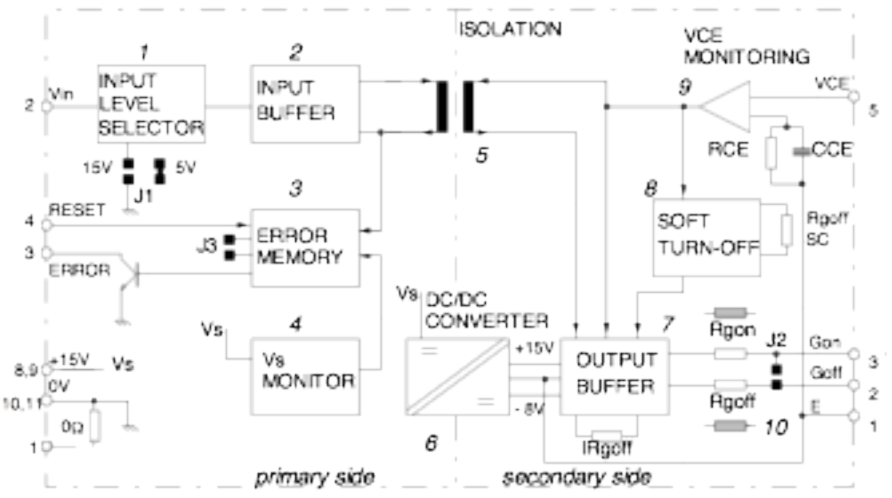
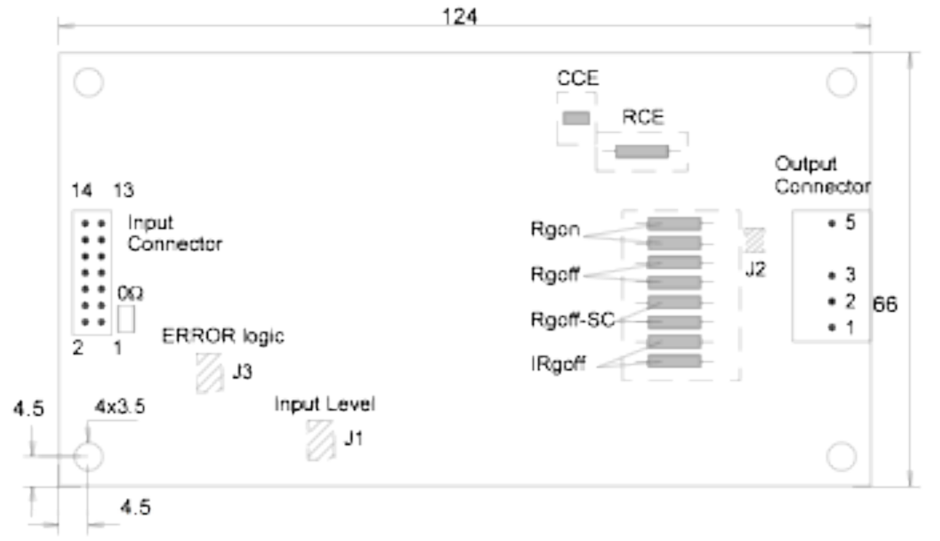
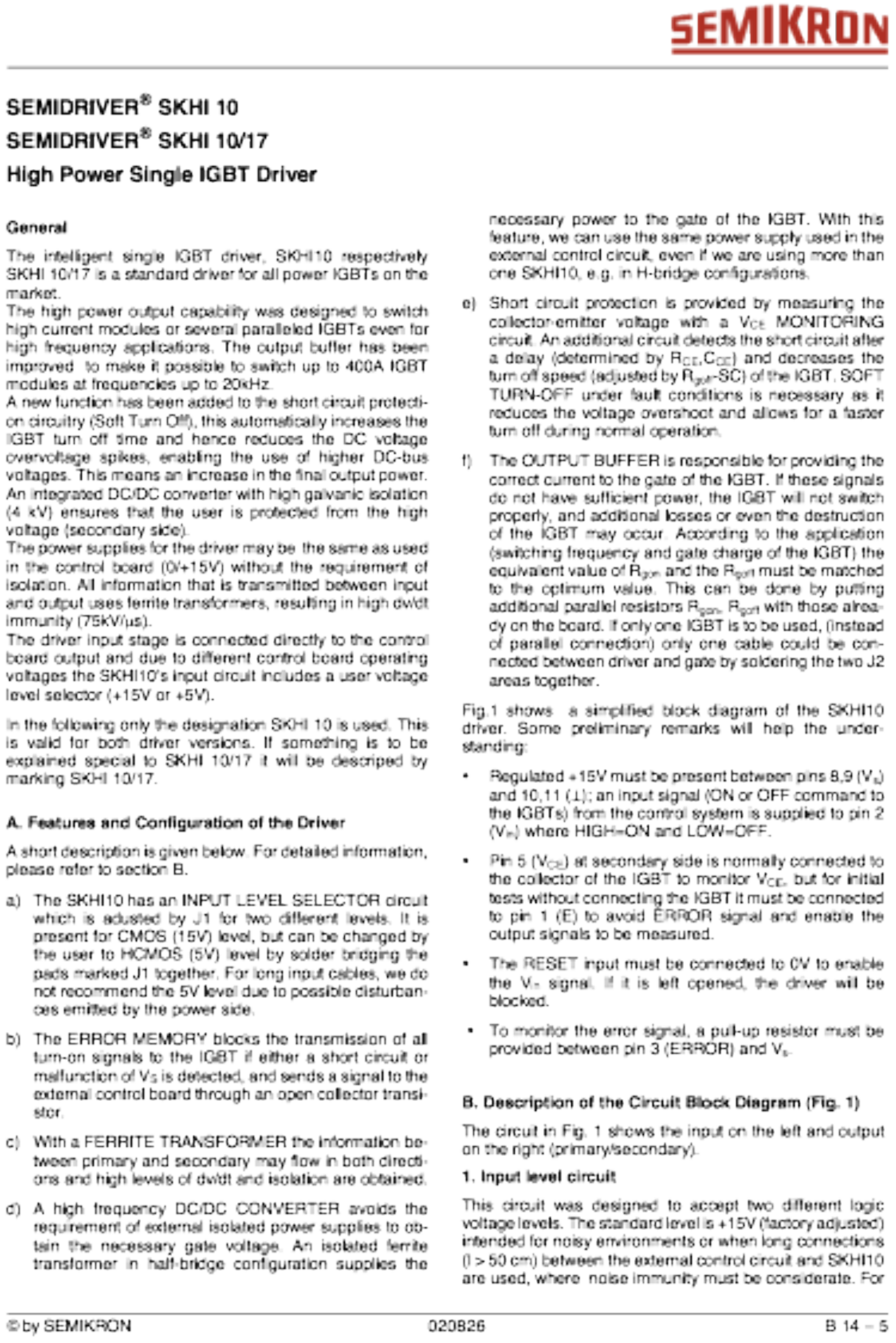


Fig.1 The numbers refer to the description on page 4, section B.



Input connector = 14 pin flat cable according to DIN 41651
Output connector = MOLEX 41791 Series (mates with 41695 crimp terminal housing and crimp terminals 7258)

Fig.2 Dimensions (in mm) and connections of the SKHI 10



SEMIKRON

SEMIDRIVER® SKHI 10
SEMIDRIVER® SKHI 10/17
High Power Single IGBT Driver

General

The intelligent single IGBT driver, SKHI10 respectively SKHI 10/17 is a standard driver for all power IGBTs on the market.
The high power output capability was designed to switch high current modules or several paralleled IGBTs even for high frequency applications. The output buffer has been improved to make it possible to switch up to 400A IGBT modules at frequencies up to 20kHz.
A new function has been added to the short circuit protection circuitry (Soft Turn Off), this automatically increases the IGBT turn off time and hence reduces the DC voltage overvoltage spikes, enabling the use of higher DC-bus voltages. This means an increase in the final output power.
An integrated DC/DC converter with high galvanic isolation (4 kV) ensures that the user is protected from the high voltage (secondary side).
The power supplies for the driver may be the same as used in the control board (0/+15V) without the requirement of isolation. All information that is transmitted between input and output uses ferrite transformers, resulting in high dv/dt immunity (75kV/µs).
The driver input stage is connected directly to the control board output and due to different control board operating voltages the SKHI10's input circuit includes a user voltage level selector (+15V or +5V).

In the following only the designation SKHI 10 is used. This is valid for both driver versions. If something is to be explained special to SKHI 10/17 it will be descriped by marking SKHI 10/17.

A. Features and Configuration of the Driver

A short description is given below. For detailed information, please refer to section B.

a) The SKHI10 has an INPUT LEVEL SELECTOR circuit which is adusted by J1 for two different levels. It is present for CMOS (15V) level, but can be changed by the user to HCMOS (5V) level by solder bridging the pads marked J1 together. For long input cables, we do not recommend the 5V level due to possible disturbances emitted by the power side.

b) The ERROR MEMORY blocks the transmission of all turn-on signals to the IGBT if either a short circuit or malfunction of V_S is detected, and sends a signal to the external control board through an open collector transistor.

c) With a FERRITE TRANSFORMER the information between primary and secondary may flow in both directions and high levels of dv/dt and isolation are obtained.

d) A high frequency DC/DC CONVERTER avoids the requirement of external isolated power supplies to obtain the necessary gate voltage. An isolated ferrite transformer in half-bridge configuration supplies the necessary power to the gate of the IGBT. With this feature, we can use the same power supply used in the external control circuit, even if we are using more than one SKHI10, e.g. in H-bridge configurations.

e) Short circuit protection is provided by measuring the collector-emitter voltage with a V_{CE} MONITORING circuit. An additional circuit detects the short circuit after a delay (determined by R_{CE},C_{CE}) and decreases the turn off speed (adjusted by R_{goff}-SC) of the IGBT. SOFT TURN-OFF under fault conditions is necessary as it reduces the voltage overshoot and allows for a faster turn off during normal operation.

f) The OUTPUT BUFFER is responsible for providing the correct current to the gate of the IGBT. If these signals do not have sufficient power, the IGBT will not switch properly, and additional losses or even the destruction of the IGBT may occur. According to the application (switching frequency and gate charge of the IGBT) the equivalent value of R_{gon} and the R_{goff} must be matched to the optimum value. This can be done by putting additional parallel resistors R_{gon}, R_{goff} with those already on the board. If only one IGBT is to be used, (instead of parallel connection) only one cable could be connected between driver and gate by soldering the two J2 areas together.

Fig.1 shows a simplified block diagram of the SKHI10 driver. Some preliminary remarks will help the understanding:

- Regulated +15V must be present between pins 8,9 (V_S) and 10,11 (⊥); an input signal (ON or OFF command to the IGBTs) from the control system is supplied to pin 2 (V_{in}) where HIGH=ON and LOW=OFF.
- Pin 5 (V_{CE}) at secondary side is normally connected to the collector of the IGBT to monitor V_{CE}, but for initial tests without connecting the IGBT it must be connected to pin 1 (E) to avoid ERROR signal and enable the output signals to be measured.
- The RESET input must be connected to 0V to enable the V_{in} signal. If it is left opened, the driver will be blocked.
- To monitor the error signal, a pull-up resistor must be provided between pin 3 (ERROR) and V_S.

B. Description of the Circuit Block Diagram (Fig. 1)

The circuit in Fig. 1 shows the input on the left and output on the right (primary/secondary).

1. Input level circuit

This circuit was designed to accept two different logic voltage levels. The standard level is +15V (factory adjusted) intended for noisy environments or when long connections (l > 50 cm) between the external control circuit and SKHI10 are used, where noise immunity must be considerate. For lower power, and short connections between control and driver, the TTL-HCMOS level (+5V) can be selected by carefully soldering the small areas of J1 together, specially useful for signals coming from µP based controllers.

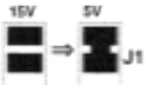


Fig.3 Selecting J1 for 5V level (TTL)

When connecting the SKHI10 to a control board using short connections no special attention needs to be taken (Fig. 4a).

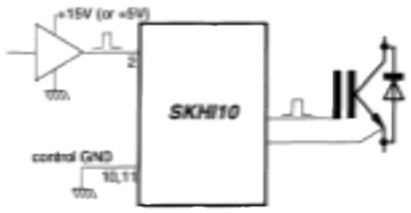


Fig.4a Connecting the SKHI10 with short cable

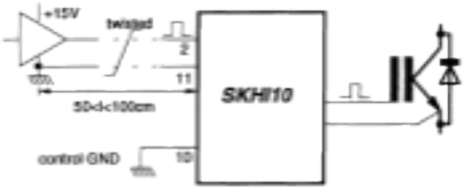


Fig.4b Connecting the SKHI10 with long cable

Otherwise, if the length is 50cm or more (we suggest to limit the cable length to about 1 meter), some care must be taken. The TTL level should be avoided and CMOS/15V is to be used instead; flat cable must have the pairs of conductors twisted or be shielded to reduce EMI/RFI susceptibility (Fig. 4b). If a shielded cable is used, it can be connected to pin 1. It is coupled to 0V through a resistor (0 Ω).

As the input impedance of the INPUT LEVEL SELECTOR circuit is very high, an internal pull-down resistor keeps the IGBT in OFF state in case the Vin connection is interrupted or left non connected.

2. Input buffer

This circuit enables and amplifies the input signal Vin to be transferred to the pulse transformer when RESET (pin 4) is LOW and also prevents spurious signals being transmitted to the secondary side.

The following overview is showing the input threshold voltages

V_{IT+} (High)	min	typ	max
15 V	9,5 V	11,0 V	12,5 V
5 V	1,8 V	2,0 V	2,4 V

V_{IT-} (Low)	min	typ	max
15 V	3,6 V	4,2 V	4,8 V
5 V	0,50 V	0,65 V	0,80 V

3. Error memory and reset signal

The ERROR memory is triggered only by following events:
- short circuit of IGBTs
- V_S-undervoltage

In case of short circuit, the V_{CE} monitor sends a trigger signal (fault signal) through the impulse transformer to a FLIP-FLOP on the primary side giving the information to an open-collector transistor (pin 3), which may be connected to the external control circuit as ERROR message in HIGH logic (or LOW if J3 is short-circuited). If V_S power supply falls below 13V for more than 0,5ms, the same FLIP-FLOP is set and pin 3 is activated. For HIGH logic (default), an external R_C must be connected preferentially in the control main board. In this way the connection between main board and driver is also checked .
If low-logic version is used (J3 short-circuited), an internal pull-up resistor (internally connected to V_S) is provided, and the signal from more SKHI10s can be connected together to perform an wired-or-circuit.

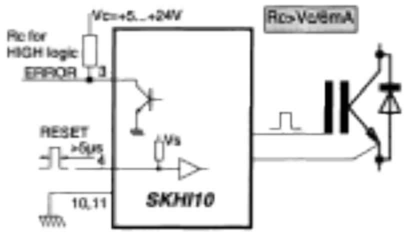


Fig.5 Driver status information ERROR, and RESET

The ERROR signal may be disabled either by RESET=HIGH (pin4) or by switching the power supply (V_S) off. The width of the RESET pulse must be more than 5µs, and in case of interrupted connection an internal pull-up resistor will act.

FAULT	RESET	ERROR [1]	V_{in}
no	0	0	enable
no	1	0	disable
yes	0	1	disable
yes	1	0	disable

1) default logic (HIGH); for LOW logic the signals are complementary

Table 1 ERROR signal truth table

The open-collector transistor (pin 3) may be connected through a pull-up resistor to an external (internal V_S for the "low-logic" version) voltage supply +5V...+24V, limiting the current to $I_{sink} \leq$ 6mA.

4. Power supply (Vs) monitor

The supply voltage V_S is monitored. If it falls below 13V an ERROR signal is generated and the turn-on pulses for the IGBT's gate are blocked.

5. Pulse transformer

It transmits the turn-on and turn-off signals to the IGBT. In the reverse direction the ERROR signal from the V_{CE} monitoring is transmitted via the same transformer. The isolation is 4 kV.

6. DC/DC converter

In the primary side of the converter, a half-bridge inverter transfers the necessary energy from Vs to the secondary of a ferrite transformer. In the secondary side, a full bridge and filters convert the high frequency signal coming from the primary to DC levels (+15V/- 8V) that are stabilised by a voltage regulator circuit.

7. Output buffer

The output buffer is supplied by the +15V/- 8V from the DC/DC converter. If the operation proceeds normally (no fault), the on- and off-signal is transmitted to the gate of an IGBT through R_{gon} and R_{goff}. The output stage has a MOSFET pair that is able to source/sink up to 8A peak current to/from the gate improving the turn-on/off time of the IGBT. Additionally, we can select I_{Rgoff} (see Fig. 2) either to discharge the gate capacitance with a voltage source (standard) or with a current source, specially design for the 1700V IGBT series (it speeds up the turn-off time of the IGBT). The present factory setting is voltage source (I_{Rgoff} = 0Ω). Using the current source I_{Rgoff}, R_{goff} must be 0 Ω.

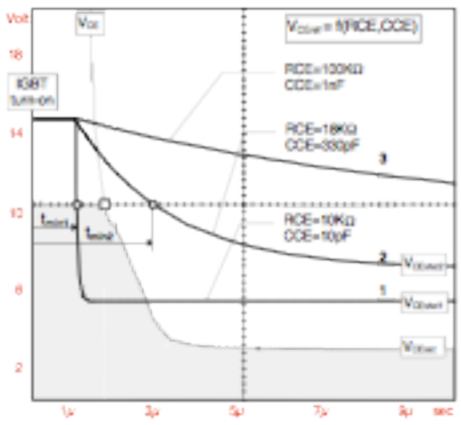


Fig.6 V_{CEref} waveform with parameters R_{CE}, C_{CE}

8. Soft turn-off

In case of short-circuit, a further circuit (SOFT TURN-OFF) increases the resistance in series with R_{goff} and turns-off the IGBT at a lower speed. This produces a smaller voltage spike (due LSTRAY x di/dt) above the DC link by reducing the di/dt value. Because in short-circuit conditions the Homogeneous IGBT's peak current increases up to 8 times the nominal current (up to 10 times with Epitaxial IGBT structures), and some stray inductance is ever present in power circuits, it must fall to zero in a longer time than at normal operation. This "soft turn-off time" can be reduced by connecting a parallel resistor R_{goff}-SC (see Fig. 2) with those already on the printed circuit board.

9. V_{CE} monitoring

This circuit is responsible for short-circuit sensing. Due to the direct measurement of V_{CEsat} on the IGBT's collector, it blocks the output buffer (through the soft turn-off circuit) in case of short-circuit and sends a signal to the ERROR memory on the primary side. The recognition of which V_{CE} level must be considered as a short circuit event, is adjusted by R_{CE} and C_{CE} (see Fig. 2), and it depends of the IGBT used. Typical values R_{CE} =18kΩ and C_{CE} =330 pF for SKHI 10 are delivered from factory (Fig. 6, curve 2). Using SKHI 10/17 the driver will be delivered with R_{CE} = 36 kΩ and C_{CE} = 470 pF from factory.

The V_{CEref} is not static but a dynamic reference which has an exponential shape starting at about 15V and decreases to V_{CEstat} ($5V \leq V_{CEstat} \leq 10V$ determinated by R_{CE}), with a time constant τ ($0{,}5\ \mu s \leq \tau \leq 1ms$ controlled by C_{CE}). The V_{CEstat} must be adjusted to remain above V_{CEsat} in normal operation (the IGBT is already in full saturation).

To avoid a false failure indication when the IGBT just starts to conduct (V_{CEsat} value is still too high) some decay time must be provided for the V_{CEref}. As the V_{CE} signal is internally limited at 10V, the decay time of V_{CEref} must reach this level after V_{CE} or a failure indication will occur (see Fig.6, curve 1). A t_{min} is defined as function of V_{CEstat} and τ to find out the best choice for R_{CE} and V_{CE} (see Fig.6, curve 2). The time the IGBT come to the 10V (represented by a „□" in Fig. 6) depends on the IGBT itself and R_{gon} used.

The R_{CE} and C_{CE} values can be found from Fig. 7 by taking the V_{CEstat} and t_{min} as input values with following remarks:

- R_{CE} > 10KΩ
- C_{CE} < 2,7nF

Attention!: If this function is not used, for example during the experimental phase, the V_{CE} MONITORING must be connected with the EMITTER output to avoid possible fault indication and consequent gate signal blocking.

10. R_{gon}, R_{goff}

These two resistors are responsible for the switching speed of each IGBT. As an IGBT has input capacitance (varying during the switching time) which must be charged and discharged, both resistors will dictate what time must be taken to do this. The final value of resistance is difficult to predict, because it depends on many parameters, as follows:

- DC-link voltage
- stray inductance of the circuit
- switching frequency
- type of IGBT

C. Operating Procedure

1. One IGBT connection

To realize the correct switching and short-circuit monitoring of one IGBT some additional external components must be used (Fig.8).

The driver is delivered with four R_g resistors (43Ω). This value can be reduced to use the driver with bigger modules or higher frequencies/lower voltages, by putting additional resistors in parallel to the existing ones.

)

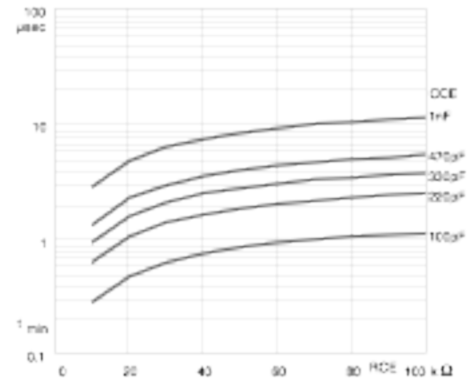


Fig.7a t_{min} as function of R_{CE} and C_{CE}

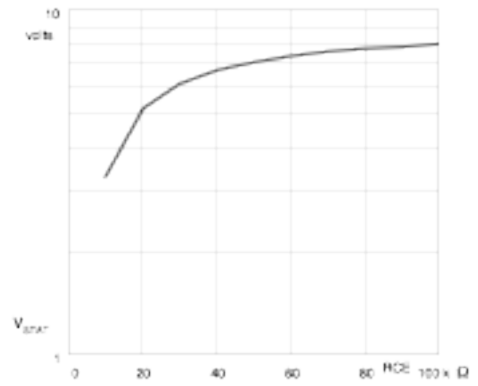


Fig. 7b V_{CEstat} as function of R_{CE}

The outputs G_{on} and G_{off} were previewed to connect the driver with more than one IGBT (paralleling). In that case we need both signals ON/OFF separately to connect additional external resistors R_{gon} and R_{goff} for each IGBT. If only one IGBT is to be used, we suggest to connect both points together through J2 (see Fig. 1 and 2). This can be done by soldering the two small pads together, which saves one external connection.

Typical component values: *)

SK-IGBT-Module	R_{Gon} Ω	R_{Goff} Ω	C_{CE} pF	R_{CE} kΩ	I_{Rgoff} Ω
SKM 75GAL123D	22	22	330	18	0
SKM 100GAL(R)123D	15	15	330	18	0
SKM 150GAL(R)123D	12	12	330	18	0
SKM 200GA(L/R)123D	10	10	330	18	0
SKM 300GA(L/R)123D	8,2	8,2	330	18	0
SKM 400GA123D	6,8	6,8	330	18	0
SKM 500GA123D	5,6	5,6	330	18	0

Table 2a 1200V IGBT@ DC-link< 700V

SK-IGBT-Module	R_{Gon} Ω	R_{Goff} Ω	C_{CE} pF	R_{CE} kΩ	I_{Rgoff} Ω
SKM 200GAL173D	8,2	8,2	470	36	0
SKM 300GA173D	6,8	6,8	470	36	0
SKM 400GA173	5,6	5,6	470	36	0

Table 2b 1700V IGBT@ DC-link< 1000V

*) Only starting values, for final optimization.

The adjustment of R_{goffSC} (factory adjusted R_{goffSC} = 22 Ω) should be done observing the overvoltages at the module in case of short circuit. When having a low inductive DC-link the module can be switched off faster.

The values shown should be considered as standard values for a mechanical/electrical assembly, with acceptable stray inductance level, using only one IGBT per SKHI10 driver. The final optimized value can be found only by measuring.

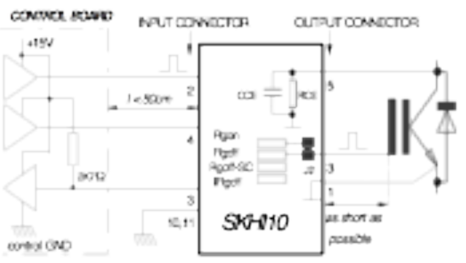


Fig. 8 Preferred standard circuit

2. Paralleling IGBTs

The parallel connection is recommended only by using IGBTs with homogeneous structure (IGHT), that have a positive temperature coefficient resulting in a perfect current sharing without any external auxiliary element. After all some care must be considered to reach an optimized circuit and to obtain the total performance of the IGBT (Fig. 9). The IGBTs must have independent values of R_{gon} and R_{goff}. An auxiliary emitter resistor R_e as well as an auxiliary collector resistor R_c must also be used.

The external resistors R_{gonx}, R_{goffx}, R_{ex} and R_{cx} should be mounted on an additional circuit board near the paralleled modules, and the R_{gon}/R_{goff} on the driver should be changed to zero ohms.
The R_{ex} assumes a value of 0,5Ω and its function is to compensate the wiring resistance in the auxiliary emitters what could make the emitter voltage against ground unbalanced.
The R_{cx} assumes a value of 47 Ω and its function is to create an average value of V_{CEsat} in case of short circuit for V_{CE}

SEMIKRON

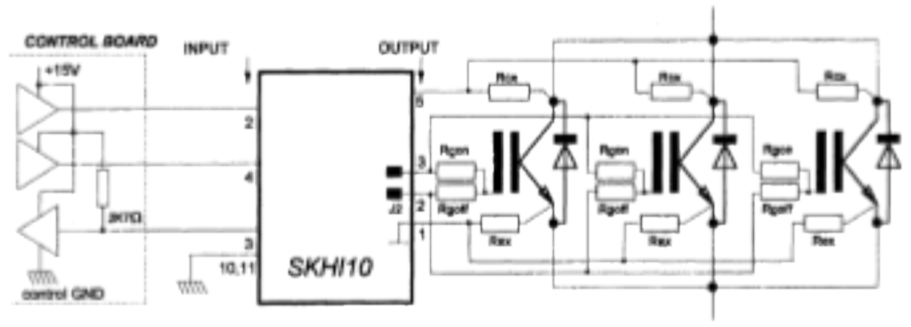


Fig.9 Preferred circuit for paralleled IGBTs

monitoring.
The mechanical assembly of the power circuit must be symmetrical and low inductive.
The maximum recommended gate charge is 9,6μC.
See also Fig.14.

D. Signal Waveforms

The following signal waveforms were measured under the conditions below:

- V_S = 15V
- T_{amb} = 25°C
- load = SKM150GAL161D
- R_{CE} = 18kΩ
- C_{CE} = 330pF
- U_{DC} = 1200V
- I_C = 100A

All results are typical values if not otherwise specified.

The limit frequency of SKHI10 depends on the gate charge connected in its output pins.
If small IGBT modules are used, the frequency could theo-

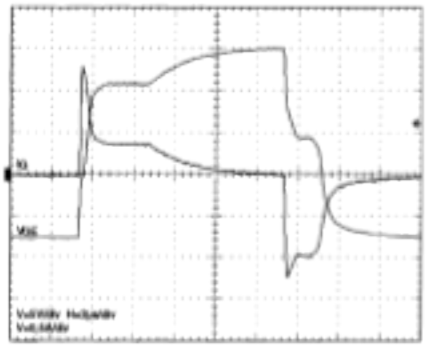


Fig.11 Output voltage (V_{GE}) and output current (I_G)

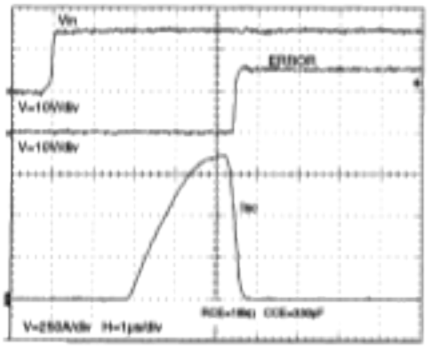


Fig.12 Short-circuit and ERROR propagation time worst-case (V_{in} with SC already present)

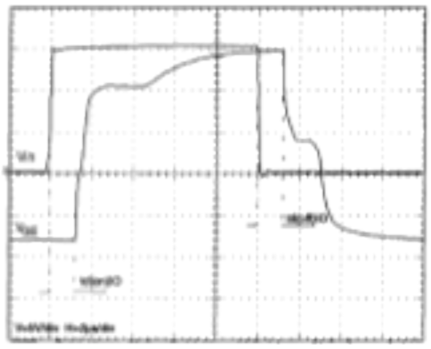


Fig.10 Input and output voltage propagation time

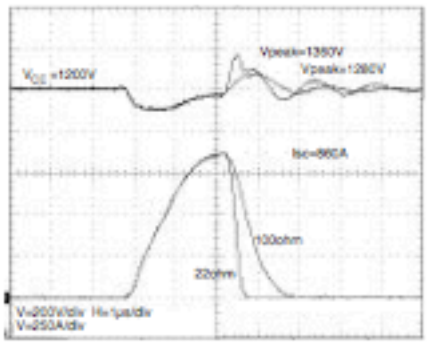


Fig.13 Effect of R_{goff}-SC in short - circuit

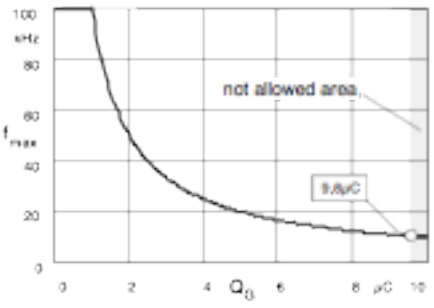


Fig.14 Maximum operating frequency x gate charge

retically reach 100kHz. For bigger modules or even paralleled modules, the maximum frequency must be determinate (Fig. 14). Q_G is the total equivalent gate charge connected to the output of the driver. The maximum allowed value is limited (9,6µC), and depends on the output internal capacitance connected to the power supply (energy storage capacitance).

E. Application / Handling

1. The CMOS inputs of the driver are extremely sensitive to overvoltage. Voltages higher than (V_S + 0,3 V) or under - 0,3 V may destroy these inputs.

Therefore the following safety requirements are to be observed:

- To make sure that the control signals do not comprise overvoltages exceeding the above values.
- Protection against static discharges during handling. As long as the hybrid driver is not completely assembled the input terminals must be short circuited. Persons working with CMOS devices should wear a grounded bracelet. Any floor coverings must not be chargeable. For transportation the input terminals must be short circuited using, for example, conductive rubber. Places of work must be grounded. The same foam requirements apply to the IGBTs.

2. The connecting leads between the driver and the power module must be as short as possible, and should be twisted.
3. Any parasitic inductance should be minimized. Overvoltages may be damped by C or RCD snubber networks between the main terminals [3] = C1 (+) and [2] = E2 (-) of the power module.
4. When first operating a newly developed circuit, low collector voltage and load current should be used in the beginning. These values should be increased gradually, observing the turn-off behavior of the free-wheeling diodes and the turn-off voltage spikes across the IGBT by means of an oscilloscope. Also the case temperature of the power module should be monitored. When the circuit works correctly, short circuit tests can be made, starting again with low collector voltage.
5. It is important to feed any ERROR back to the control circuit to switch the equipment off immediately in such events. Repeated turn-on of the IGBT into a short circuit, with a frequency of several kHz, may destroy the device.

For further details ask SEMIKRON

Nr.11224040