Top-down methodology employing hardware description languages (HDLs) for designing digital control in power converters

L. Laguna, R. Prieto, J. Oliver and J. Cobos
Centro de Electrónica Industrial (CEI)
Escuela Técnica Superior de Ingenieros Industriales
Universidad Politécnica de Madrid
José Gutiérrez Abascal 2, Madrid, Spain, 28006
Tel.: +34 91 336 31 92, Fax.: +34 91 564 59 66
E-mail: cei@upm.es

Abstract

This paper presents a research line oriented to develop methodologies that takes advantage of hardware description languages in order to simplify the design of power converters that employ digital control techniques. The methodology focuses on setting the adequate communications among subsystems in order to simplify the change of the levels of abstraction of the subsystem's models (from the conceptual level to the actual electric + synthesizable code). Changing the level of abstraction in the design process pretends: first to provide useful models at early designing steps; second, to optimize the simulation of the system, and at same time optimize the verification step.

1. Introduction

The advantages of using digital control techniques in power converters has been illustrated in many publications [1][2][3]. Most of the design flows that have been presented involve the use of Matlab to design the control and Simulink to simulate it. This approach has the advantage that Matlab and Simulink are integrated, so it is really easy to design and implement a model in the simulator. However Simulink is not an electric circuit simulator. Several solutions to simulate electric circuits in Simulink have been presented [4]. On the other hand, specialized electrical simulators are not suitable to develop and simulate digital circuits in early stages of the design process because there are some lacks in flexibility when developing behavioral models. The use of hardware description languages (VHDL, Verilog) in the design of digital circuits is a well known technique. These languages have really nice features to design systems using top-down methodologies. Now it is possible to simulate systems of various disciplines employing the same tools and methodologies with the extension of these languages (VHDL-AMS, Verilog-AMS), [5][6]. In the case of power converters with digital control, it is possible to simulate the actual code implemented in the FPGA or ASIC in conjunction with the electrical model of the converter. The application of a bottom-up methodology in the design of converters with digital control is still good enough. However, as shown in [7], if the system becomes more complex this approach presents problems.

2. The design methodology

The proposed design methodology can be resumed in the following steps:

1. Division of the system into subsystems
2. Definition of the nature of communication ports for each subsystem
3. Subsystems models development and interface definition
4. Settling of suitable configurations of the system
5. Bottom-up verification
6. Top-down verification

2.1. Division of the system into subsystems

In any top-down design methodology, in order to simplify the problem the system is continuously divided into small units or subsystems. Different approaches can be used to divide the system. The general recommendation is to divide the system in the most intuitive way for the designer. In the example, the system is divided in five major subsystems: converter, digital control (FPGA), sensors (ADCs) and actuators (PWM) as shown in figure 1. This division was made based on actual physical construction. It is also separated into analog and digital parts.

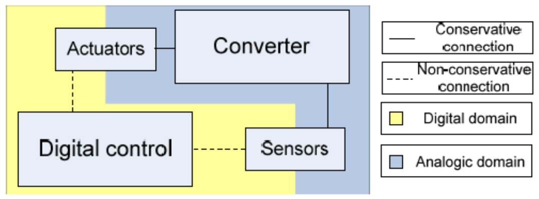

Figure 1. Subsystem division of the multiphase converter

2.2. Definition of the nature of communication ports for each subsystem

Once the system has been subdivided, it is necessary to establish how the subsystems are communicated (its interface with the other models). In most HDLs it is possible to choose between two types of ports to communicate subsystems: conservative and non-conservative ports (more details can be found in [8]). In general it is recommended to set the kind of port according to the physical port nature. This means: if the port is electrical, use a conservative port. And if the port is a signal, use a non conservative port.

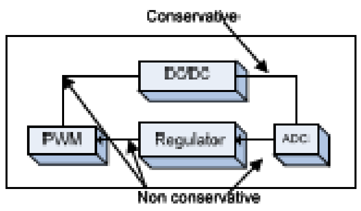

Figure 2. Conservative and non-conservative ports definition of the system

In the example (figure 2), the converter has only conservative ports because it only has electrical connections. The digital control only has non-conservative ports because it sends and receives only digital signals. In the case of doubts about which kind of port should be used for a subsystem, a conservative port should be used because conservative ports can transmit non conservative data.

A correct definition of the port type eases the process of changing the level of abstraction of the models, for example, most converters can be modeled in a high level of abstraction with a linear transfer function which its interface has only non-conservative ports, this model can not be connected with other electric models that have conservative ports (figure 3).

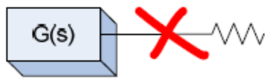

Figure 3. A transfer function can not be connected to a resistance, they have different ports.

2.3. Subsystems models development and interface definition

As shown in the previous example if the ports of two models are not of the same type, these models can not be connected. In order to connect these models, an adapter should be defined. Once the interface to a model is defined, internally the model can be described in any way, but an adapter should be defined to connect the model to the outside world. These approach has a big advantage: there could be defined models with different levels of abstraction for the same component.

The objectives of developing models with different levels of abstraction are:

- Simpler models are more useful than complete and detailed models in early stages of the design. For example: when designing a control with linear systems theory, a transfer function of the plant is useful than a detailed physical model.
- Architecture level errors (interconnection, communication and concept errors) are easily found in abstract models [7].
- Abstract models are faster to simulate. For example: significant improves in the simulation time can be achieved (10x of more) when using averaged models of SMPSs.

The different models that can be used in the example are shown in figure 4. For the DC/DC converter: transfer function, averaged model, switching model. The digital regulator: discrete transfer function, fixedpoint transfer function and synthesizable code. The PWM can be: synthesizable or behavioral. And in this example only a behavioral model is used in the ADC.

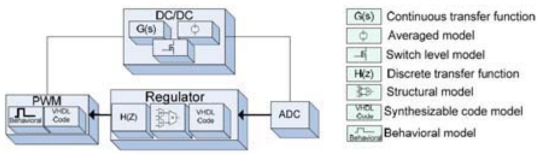

Figure 4. Different models of each subsystem (different level of abstraction)

The problem of having defined the kind ports in a subsystem is that not all the different models of subsystem can communicate through the port defined. For example when using the transfer function model (figure 5.a) it is necessary to put an adapter in order to connect the non-conservative ports with the conservative ones. In contrast to the switched model that has a direct connection to the conservative ports (figure 5.b).

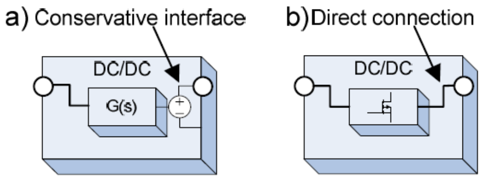

Figure 5. Adapters for the different models of the DC/DC converter a) conversion from non- conservative to conservative. b) no conversion needed.

In fact, this approach has more advantages than disadvantages because having defined the interconnection of the models is very simple to switch among the different levels of abstraction of a model without affecting the way this communicates with other subsystems. This allows defining different configurations for the complete system, for example: it is possible to set a high level abstract model of the system setting all subsystems with the highest level of abstraction. This point will be more explained in the next section.

2.4. Settling of suitable configurations of the system

The complete model of the converter (that in this case is called system) may be a subsystem of another major system. Therefore is needed to establish the configurations (different models with different levels of abstraction) that the complete converter will provide to its container system. This is illustrated in figure 6.

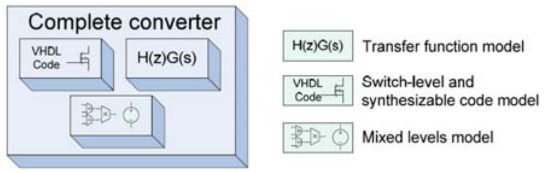
Complete converter
VHDL Code
H(z)G(s)
H(z)G(s) Transfer function model
VHDL Code
Switch-level and synthesizable code model
Mixed levels model

Figure 6. Different configurations for the complete converter system. From transfer function model to switch level + synthesizable code.

When setting the level of abstraction of the subsystems, there are some configurations that are more suitable than others. This is due that some configurations may not provide additional information when these are simulated. Two examples of suitable configurations are shown in figure 7. Figure 7.a shows the higher level configuration of the example, where all models used are transfer functions. Figure 7.b shows the most real; in this case all models have switch-level information.

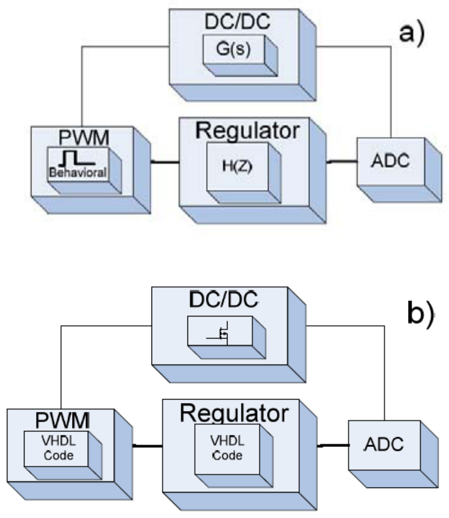

Figure 7. Two suitable configurations. a) continuous and discrete transfer functions. b) synthesizable code and electric model

2.5. Bottom-up verification:

The objective of the bottom-up verification is to assure that a subsystem behaves as it is assumed from the point of view of the system. In hardware description languages it is possible to define sub-tests for a given model. For example, in order to validate the different models of the DC/DC converter a test-bench can be defined that simulates a load step, the three models must provide the same result in a defined range. This situation is illustrated in figure 8.

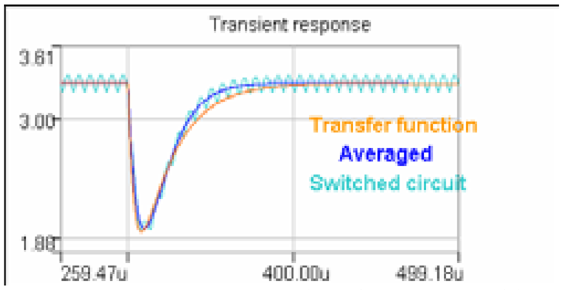

Figure 8. Bottom up verification of the DC/DC converter models with a transient response.

This eases the verification of every model when changes are made, and at the same time this allows to isolate the problems that may exist in the subsystem before using it in its container system. These test scenarios can be easily defined with most HDLs.
Even if all the parts of a system work correctly, is not a guaranty that this parts when connected will work properly together.

2.6. Top-down verification

Simulations must be efficient in terms of “time spent performing the simulation” over “information obtained”. Top-down verification allows to maximize the simulation efficiency because the system is validated starting with the fastest models (most abstract models) and when the systems fulfills the requirements the next abstraction level is validated. If the system is validated starting form the lowest abstraction model it could take a lot of time to simulate.

3. Advantages of text models

Working with HDL models has some advantages over the schematic representations. Some of those advantages are:

- Using a version control system like Subversion (SVN) can help to track the changes in the model. That’s a very useful tool that has been used by the software community for years, in the development of large projects.
- In many cases, a text representation of the model is easy to read and modify, especially in the description of the digital parts, where the gate diagrams in early design steps are almost obsolete since many years.
- Most of the HDLs allow defining automatic tests, and this test format is easier to represent as a text file.

4. Conclusions

A methodology for the design of systems like power converters that employs digital controllers was presented. This methodology takes advantage of the AMS hardware description languages and can be used in the design of large systems. Thanks to the capabilities that hardware description languages (HDL) provide, it is possible to describe abstract models in early designs to validate the concepts, and after, describe detailed models for depth verifications. One of the most important advantages of HDLs is that low level synthesizable code can be simulated with low level electrical models. Even when the models are stored as code, there is no need to coding thanks to the graphical environments for schematic capture. The code can be recoded and tracked with version a control system, which is especially suitable when working in teams; this provides a safe and organized way to work with large code.

5. References

[1] Miao, B.; Zane, R.; Maksimovic, D, “Impact of digital control in power electronics” Power Electronics Specialists Conference, 2004. PESC 04. 2004 IEEE 35th Annual, Volume 5, 20-25 June 2004 Page(s):3728 - 3733 Vol.5

[2] Feng, G.; Meyer, E.; Liu, Y.-F. “A New Digital Control Algorithm to Achieve Optimal Dynamic Performance in DC-to-DC Converters” Power Electronics, IEEE Transactions on, Volume 22, Issue 4, July 2007 Page(s):1489 – 1498

[3] Al-Atrash, H.; Batarseh, I. “Digital Controller Design for a Practicing Power Electronics Engineer” Applied Power Electronics Conference, APEC 2007 - Twenty Second Annual IEEE Feb. 2007 Page(s):34-41

[4]http://www.mathworks.com/products/simpower/, SimPowerSystems, July/2007

[5] Pecheux, F.; Lallement, C.; Vachoux, A. “VHDL-AMS and Verilog-AMS as Alternative Hardware Description Languages for Efficient Modeling of Multidiscipline Systems” Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on

Volume 24, Issue 2, Feb. 2005 Page(s):204 – 225

[6] Christen, E.; Bakalar, K. “VHDL-AMS A Hardware Description Language for Analog and Mixed-Signal Applications” Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on, Volume 46, Issue 10, Oct. 1999 Page(s):1263 – 1272

[7]http://www.designers-guide.org/Design/top-down.pdf “Top-Down Design and Verification of Mixed-Signal Circuits”, July/2007

[8] Peter J. Ashenden, Gregory D. Peterson and Darrell A. Teegarden, “The System Designer's Guide to VHDL-AMS” Morgan Kaufmann Publishers, September 2002, ISBN 1-55860-749-8