A METHODOLOGY FOR THE DEVELOPMENT OF THE DIGITAL CONTROL OF STATIC CONVERTERS USING FPGA

Michel Santana, Enio Roberto Ribeiro, Robson Luiz Moreno
Itajubá Federal University
Av. BPS 1303 - Bairro Pinheirinho - Itajubá - Minas Gerais - Brazil
email: {michel;enio;moreno}@iee.efei.br

Abstract - **This article describes a methodology for the development of the digital control of power converters, where the whole structure – converter and control system – can be modeled in the same environment through the hardware description language VHDL. This language allows a behavioral description for the whole structure, so that the control system can be simulated, tested and validated without the need of a prototype. The portability of the VHDL language allows the implementation of the digital control code in an ASIC or FPGA. Simulations results show the feasibility of the methodology offering interesting possibilities in power converter control.**

Keywords - **ASIC, Static Converter, Digital Control, FPGA, VHDL-AMS.**

I. INTRODUCTION

Currently there are many types of power converters being used and all of them require a control system for proper operation. The control system can be implemented with analog circuits [1]. They meet the needs of simple control systems (single input single output) and even complex control systems (two or more controlled variables), and are attractive due to their simplicity of use and low cost. While analog based systems have proven successful, several reasons make digital control attractive.

Another alternative is the use of digital circuits to implement the converter control. They have been showing improvements in performance and reduction of costs. In general, the digital control systems use Digital Signal Processors (DSP) due to their mathematical resources and the auxiliary subsystems (timer module, PWM generator, etc) [2]. One disadvantage of DSP is its sequential operation, in which the instructions are executed sequentially, although the subsystems add up the simultaneous processing capability.

Digital control allows for the implementation of more functional control schemes. Digital circuits are potentially less susceptible to noise and parameter variations.

In order to meet the demand on simultaneous processing, the digital control systems are implemented using Field Programmable Gate Array (FPGA) or Application-Specific Integrated Circuit (ASIC) [2]. They are capable of performing simultaneous processing and can operate at high frequencies. These features allow the implementation of control algorithms with simultaneous procedures. The control algorithm must be properly developed to produce an object suitable to be programmed on a FPGA. This requires that both the converter and the control system be developed on the same environment, which unfortunately is not a favorable aspect on the use of FPGA.

This article presents a methodology to overcome this unfavorable aspect on the use FPGA. The methodology describes an integrated approach to build, test and validate the complete system (converter and control system), thus producing an object suitable to be programmed on a FPGA.

The control systems developed for FPGA avoid the arithmetic operations, very common on DSP, in favor of combinational, sequential and conditional operations that are more suitable for FPGA [2,3].

II. GENERAL SYSTEM

The proposed methodology will be developed using the Buck converter, with power factor correction [1], as shown in Figure 1. The goal of the converter control is the output voltage regulation in accordance to the input voltage. The converter signals to be sampled are: the input current, the rectified voltage and the output voltage. An analog or a digital system can be controlled using these signals.

The objectives of the methodology are the development, test and validation of a digital control system, as well as the generation of a programming code in hardware description language suitable to be implemented on a FPGA. This phase should be implemented on a single environment capable of integrating the whole structure – the converter and the control system.

Many circuit description models are available and they can be used in softwares, such as ORCAD®, Matlab®, among others. One option, which have been widely chosen [4,5,6], is the use of the Very High Scale Integration Circuit Hardware Description Language (VHDL). This language allows the digital controll integration into an FPGA, and increases the solution portability, once the final code can be easily transported to any manufacturer's FPGAs. Using the VHDL language, in particular the VHDL-AMS (Analog and Mixed-Signals), analog and digital circuits can be described by their behavior.

This aspect facilitates the integration of components of a variety of features (diodes, A/D converters, digital circuits, etc) on the same environment. The VHDL-AMS meets the need of description and integration of analog and digital circuits in the same environment, and it was standardized in 1999 [7,8].

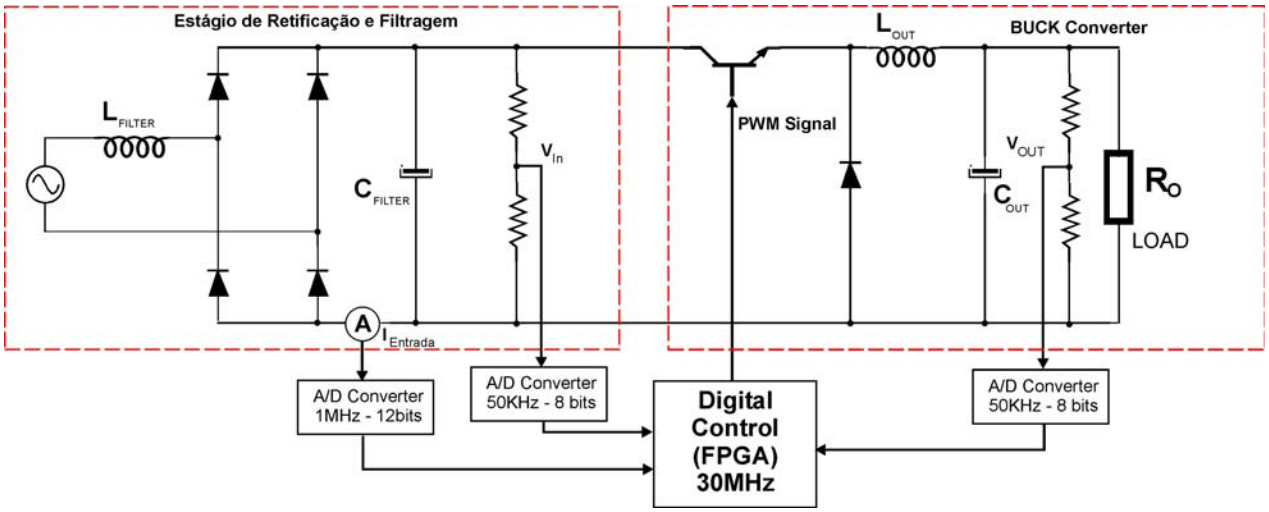


Fig. 1. Buck Converter using digital control implemented on a FPGA.

III. METHODOLOGY

The methodology consists in the construction of the whole power system – converter and the digital control system – from its description, simulation and validation in a single environment. The system is composed of distinct elements, as shown in Figure 1, that will be described separately, by their behavior and by the description language. After completing all the elements, the system will be described by a structural description [7,8].

3.1. Modeling of the components

The VHDL-AMS is used to model each element, since it allows integration and portability to the code. The elements that compose the conversion system will be evaluated in analog (capacitor, inductor, resistor, etc), digital (logic circuits, digital pulse generators, etc) or mixed mode (A/D Converters, switches, etc). In order to describe a component in a hardware description language, it is necessary to define its inputs, outputs and its behavior.

The detailing or precision level of a component can be easily adapted to the system's development requirements, using most detailed component's description. As an example, one switch can be modeled like a simple conditional test, discarding all internal resistance and capacitance, making easier the simulation process, which is responsible for the system's digital control development [5,6]. If a high precision on the analog components behavior becomes necessary, you can add to the original code the desired changes [9,10]. At this point, some of the most common components used on converters modeling are going to be presented as a sample of the language description facility. These models uses simplified description of the components behavior.

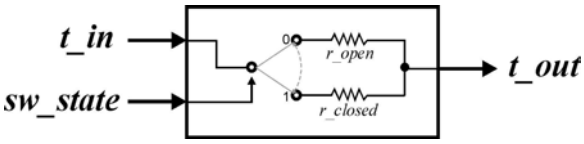
a) Modeling mixed mode components (analog-digital) - The mixed mode components offer the capability of integrating the analog and the digital parts of the system in the same structure. The switch of the Buck converter presents this characteristic, since it has analog inputs and outputs, and it is digitally activated. The switch is described as a resistance whose value is altered according to the logic value applied to its digital input, as presented in Figure 2. The model of a component described in the VHDL hardware description language (see Figure 2c) is basically composed of three parts:

- The definition of the libraries used in its description, lines 1 to 3 *(library);*
- The identification and characterization of its inputs, outputs and atributes, lines 4 to 10 *(entity)*;
- The behavioral or structural description, lines 11 to 26 *(architecture)*.

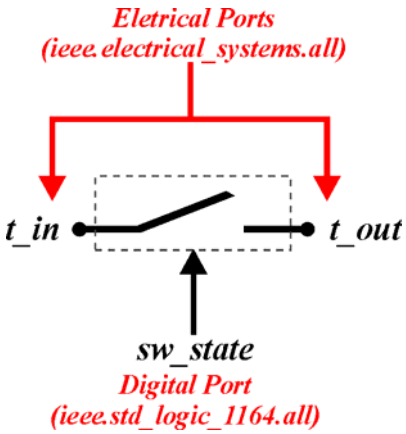
From this description it is possible to observe that terminals *t_in* and *t_out* are defined as electrical values (line 9), and the switch operation is defined as a digital value (line 8). This property allows the feature of external control to the component, which can be obtained from a PWM pulse generator or from a digital control loop, which can be described in VHDL. Another component that has this feature is the analog/digital converter that can also be modeled by the hardware description language [11].

b) Modeling of analog components - In order to implement the converter, it is necessary to describe, in VHDL-AMS, the other elements, such as power supplies, resistors, inductors, etc. Table I shows the behavioral description of few elements used in converters. The references to libraries and the characterization of elements were omitted from the description.

c) Digital components modeling - The digital control system is a complex structure composed of logic and arithmetic operators. Its implementation becomes easy from the modularization of logic blocks, such as logic operators, adders, shifters, comparators, etc. A PWM pulse generator is classified as a digital component, since its output presents a signal of logic value. Figure 3 presents the modeling of this component. Lines 5 to 8 present the behavioral description of the pulse width for the high level (*t_up*) and the low level (*t_down*). These values can be inserted by the user through the structural description, which will be analyzed on the next section.



(a) Switch signals.



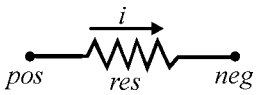
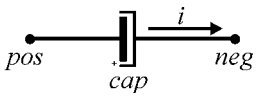
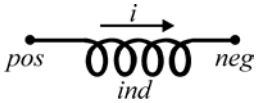
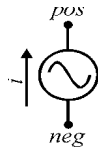
(b) Characterization of Input and Output types.

```
1.  library ieee;
2.  use ieee.electrical_systems.all;
3.  use ieee.std_logic_1164.all;

4.  entity switch_digital is
5.  generic (r_open     : resistance := 1.0e6;
6.           r_closed   : resistance := 1.0;
7.           transient_time     : real      := 1.0e-6);
8.  port    (sw_state          : in std_logic;
9.           terminal t_in, t_out    :    electrical);
10. end entity switch_digital;

11. architecture behav of switch_digital is
12. signal   r_signal : resistance := resistance_open;
13. quantity r        : resistance;
14. quantity v across i through t_in to t_out;
15. begin
16.   switch_state: process (sw_state)
17.   begin
18.     if (sw_state'event and sw_state = '0') then
19.       r_signal <= r_open;
20.     elsif (sw_state'event and sw_state = '1') then
21.       r_signal <= r_closed;
22.     end if;
23.   end process switch_state;
24.   r == r_signal'ramp(transient_time, transient_time);
25.   v == r * i;
26. end architecture behav;
```

(c) VHDL-AMS description.

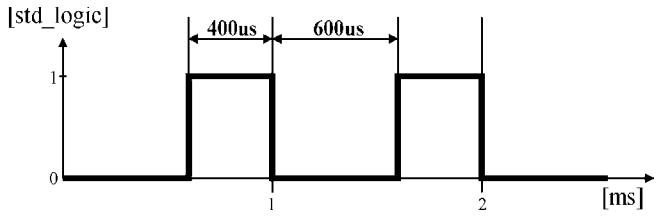
Fig. 2. Modeling of the switch.

TABLE I
VHDL-AMS behavioral description of analog elements

Component	VHDL Behavior	Comments
	*architecture behav of resistor is* *quantity v across i through pos to neg;* *begin* ***v == res * i;*** *end behav;*	*The behavior of the resistor is given by its fundamental equation:* ***v = res * i****, where res is the resistance value in ohms, to be specified by the user.*
	*architecture behav of resistor is* *quantity v across i through pos to neg;* *begin* ***i == cap * v'dot;*** *end behav;*	*The capacitor fundamental equation is:* ***i = cap * dv/dt****, where cap is capacitance value given in Faraday, to be specified. The directive 'dot describes the behavior of the derivative.*
	*architecture behav of resistor is* *quantity v across i through pos to neg;* *begin* ***v == ind * i'dot;*** *end behav;*	*The inductor fundamental equation is :* ***v = ind * di/dt****, where ind is the inductance in Henry, to be specified. The directive 'dot describes the behavior of the derivative.*
	*architecture behav of v_sine is* *quantity v across i through pos to neg;* *begin* ***v == amplitude * sin(math_2_pi * freq * NOW);*** *end behav;*	*The sinusoidal voltage is described by its fundamental equation:* ***v(t) = amplitude * sin(2 * π * freq * t)****. The amplitude and the frequency (freq) are specified by the user. The directive NOW emulates the behavior as function of the time.*

```
1.  architecture behav of pwm_generator is
2.    begin
3.      CreateClock: process
4.        begin
5.          clk_out <= '0';
6.          wait for t_down ;
7.          clk_out <= '1';
8.          wait for t_up ;
9.      end process CreateClock;
10  end behav;
```

(a) VHDL description.



(b) PWM Generator waveform.

Fig. 3. Modeling of the PWM generator.

3.2. Structural description

The structural description is composed of instantiations of components which have already been validated. Figure 4a presents a circuit that used the formerly described components. Figure 4b shows the result of the simulation of a circuit. The signal TEST.CO.V represents the sinusoidal output signal. The digitally described signal TEST.CLK_PWM is the switch control signal. The signal TEST.C3.V is the voltage between *node2* and *ground*. From the simulation result presented in Figure 4b it is possible to validate the structural description. The structural description is presented in Figure 5.

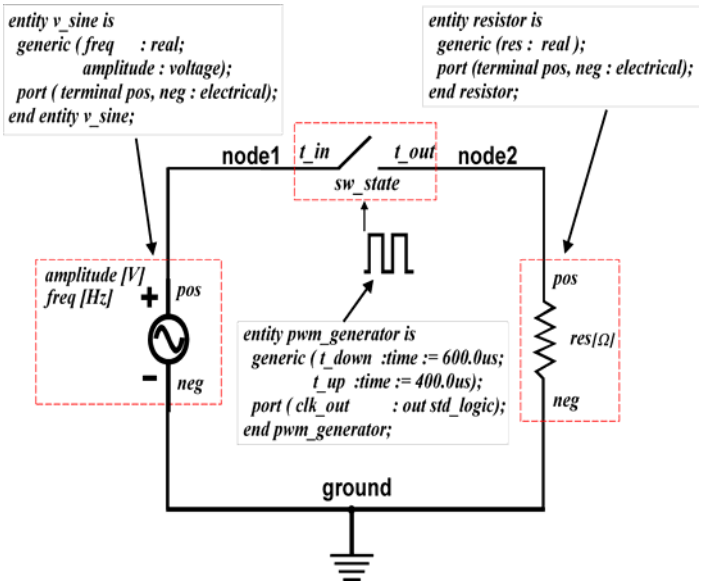
All components utilized in the structure must have already been inserted in the environment, as exemplified in line 4 of this procedure. The instantiation of the used components is performed through the processes (lines 11, 14, 17 and 20). Signals (lines 8 and 9) are used in order to integrate the components on the same environment. The signals allow the connectivity of the blocks on the structural description and are similar to the nodes of an electric circuit.

The attributes of the signals used in the description are set through the VHDL directive *generic map*. The connectivity is performed by the *port map* using temporary signals [7,8].

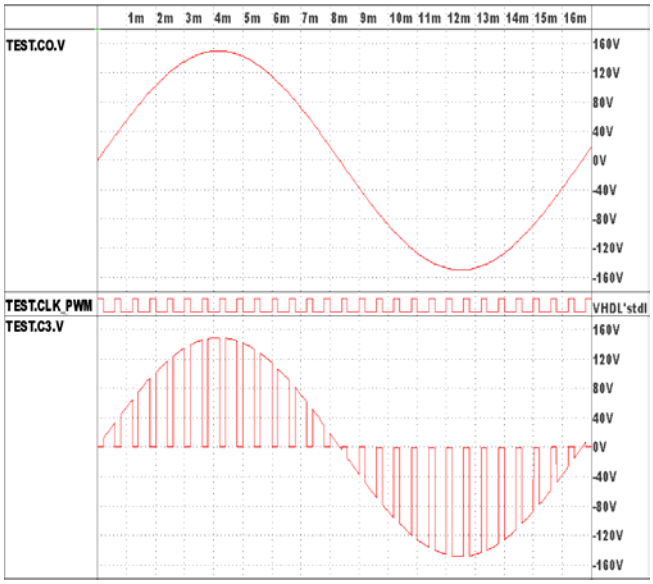
IV. CONVERTER SIMULATION

Figure 6 present the circuit of a Buck converter. Figure 7 presents the SMASH® simulation results, in order to demonstrate the feasibility of the methodology. The signals TEST.C0.V, TEST.C0.I and TEST.C10.V represent, respectively, the inductor current, the capacitor current and the voltage at the output of the Buck converter.

From the results shown in Figure 7, it can be verified that the converter structural description is valid and the circuit has a settling time of approximately 16 ms according to the parameters described in Figure 6.



(a) Circuit example.



(b) Simulation results.

Fig. 4. A structural description example.

```
1.  library IEEE;
2.  use IEEE.electrical_systems.all;
3.  use IEEE.std_logic_1164.all;
4.  use work.all;

5.  entity Test is
6.  end Test;

7.  architecture Top of  Test  is
8.  signal  clk_pwm : std_logic;
9.  terminal node1, node2: electrical;
10. begin
11. C0:  entity v_sine (behav)
12.       generic map (amplitude => 150.0, freq => 60.0)
13.      port map(pos => node1, neg => ground);
14. C1:  entity pwm_generator(behav)
15.       generic map (t_down => 600.0us, t_up => 400.0us)
16.       port map(clk_out => clk_pwm);
17. C2:  entity switch_digital(behav)
18.       generic map (r_open => 0.001, r_closed => 1.0e6, transient_time => 1.0e-6)
19.       port map(sw_state => clk_out, t1=> node1, t2=> node2);
20. C3:  entity resistor(behav)
21.      generic map (res => 100.0)
22.      port map (t1 => node2, t2 => ground);
23. end Top;
```

Fig. 5. Structural description of circuit in Fig. 4.

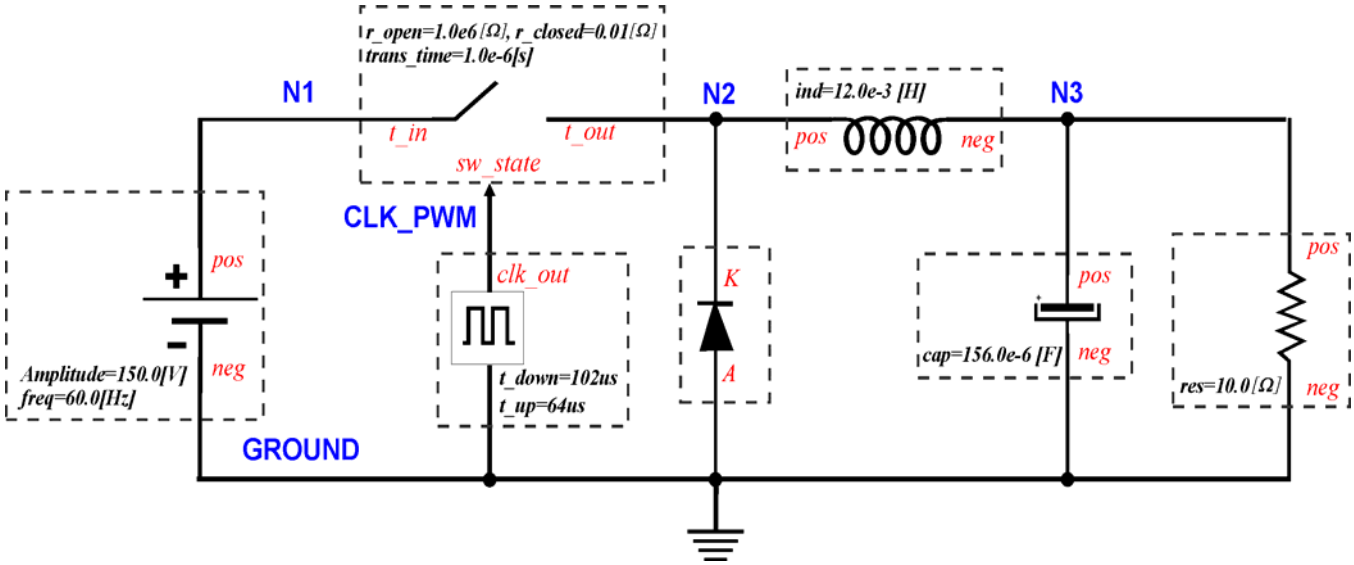
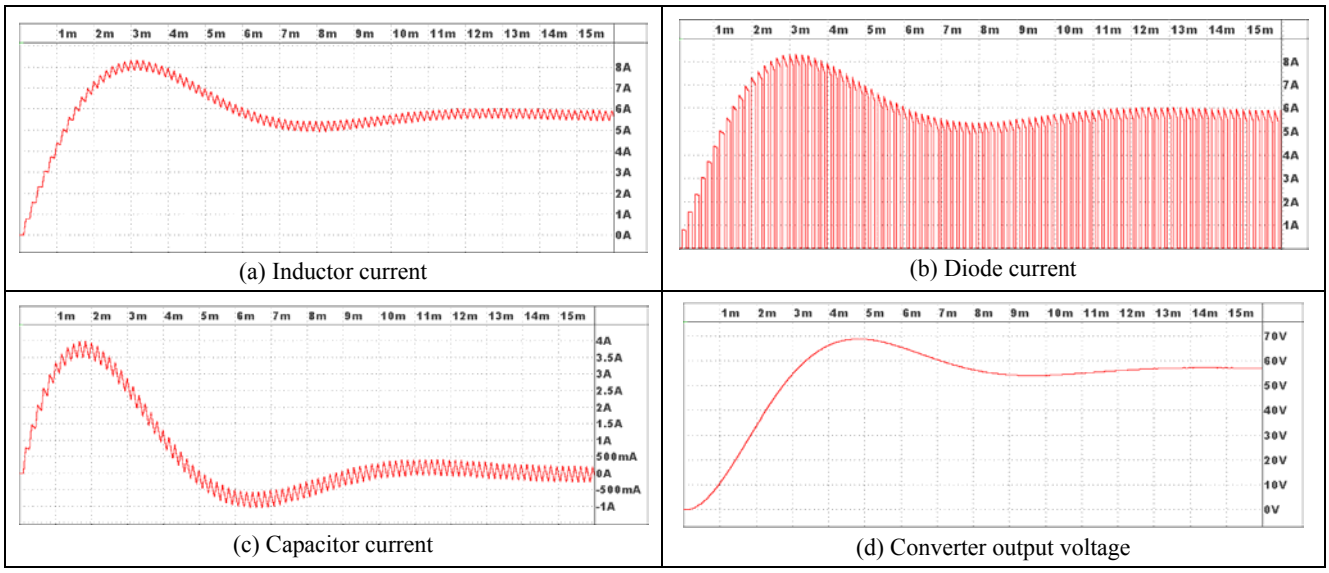


Fig. 6. DC/DC converter.

IV. DIGITAL CONTROL

To the digital control development, some digital components creation, which makes the implementation possible, becomes needed. Making use of combinational circuits, it is possible to have two or more processes running at the same time, what allows easy implementation of protection to max input currents, max output voltage, max duty cycle, etc [2]. The most commonly used components on controlling are adders, accumulators, multipliers, and comparators, because they can emulate the classical controllers' behavior, which make use of integration and derivation techniques to implement the controlling strategy. The comparators use, with A/D converters, makes possible the implementation of circuit's needed protection. Table II shows the behavioral description of few elements used in digital control.



(a) Inductor current

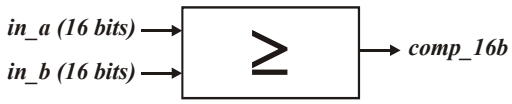
(b) Diode current

(c) Capacitor current

(d) Converter output voltage

Fig. 7. Buck converter simulation results.

TABLE II
VHDL-AMS behavioral description of digital elements

Component	VHDL Behavior
	*architecture behav of mult is* *begin* *process* *variable temp,tempa,tempb : std_logic_vector(15 downto 0);* *begin* *tempa := "00000000" & in_a;* *tempb := "00000000" & in_b;* *temp := unsigned(tempa) * unsigned(tempb);* *out_mult <= temp(15 downto 0);* *end process;* *end behav;*
	*architecture behav of sumacc is* *begin* *process* *variable tempa, tempb : std_logic_vector (15 downto 0);* *begin* *wait until clk'event and clk = '1';* *if clr_acc='1' then* *out_acc <= (others => '0');* *tempa := (others => '0');* *else* *tempb := "00000000" & in_a;* *tempa := (unsigned(tempa)) + (unsigned(tempb));* *out_acc <= tempa (15 downto 0);* *end if;* *end process ;* *end behave;*
	*architecture behav of comp_16b is* *begin* *compare : process (cmp)* *variable in1_hold : std_logic_vector (15 downto 0);* *variable in2_hold : std_logic_vector (15 downto 0);* *begin* *in1_hold := in_a;* *in2_hold := in_b;* *if cmp'event and cmp = '1' then* *if in1_hold >= in2_hold then* *eq <= '1';* *else* *eq <= '0';* *end if;* *end if;* *end process;* *end architecture behav;*

V. CONCLUSION

This article presented a methodology for the design of a power converter system. The fundamentals necessary for the description of circuits using VHDL-AMS were presented. The methodology allows description, test and validation the digital control code, in VHDL-AMS, along with the other converter components. By using this methodology it is possible to develop the whole structure of the power conversion, considering the responses of the iteration between the control system and the converter. This reduces time and costs at the experimental phase. The VHDL code obtained at the end of the process can be transferred to any FPGA or ASIC programming environment. The Physical implementation, as well as the choose of an economically viable controlling system, comparing to the analog controllers, are future works proposals.

REFERENCES

[1] Barbi, F. Pöttker de Souza, "A Unity Power Factor Buck Pre-Regulator with Feedforward of the Output Inductor Current", *IEEE Applied Power Electronics Conference (APEC)*, 1999.

[2] A. de Castro, P. Zumel, O. García, T. Riesgo and J. Uceda, "Concurrent and Simple Digital Controller of an AC/DC Converter with Power Factor Correction", *IEEE Trans. Ind. Electron.*, vol. 46, pp. 3-12, Fevereiro de 1999.

[3] Wu, A.M.; Jinwen Xiao; Markovic, D.; Sanders, S.R., **"**Digital PWM control: application in voltage regulation modules**",** *Power Electronics Specialists Conference*, vol. 1, pp. 77 – 83, 1999.

[4] Lukasz Starzak, Andrzej Napieralski, Jean-Jacques Charlot, "VHDL-AMS: A Competitor for SPICE", *Modeling of Semiconductor Devices*, pp. 353-356, TCSET 2002, Frebruary 2002.

[5] Teresa Riesgo, Yago Torroja, Eduardo de la Torre, "Design Methodologies Based on Hardware Description Languages", *IEEE Transactions on Industrial Electronics*, pp. 3-12, Vol. 46, No. 1, February 1999.

[6] Ahmed Fakhfakh, H. Levi, N. Milet-Lewis, Y. Danto. "Behavioral modeling of analogue and mixed integrated systems with VHDL-AMS for RF applications", *XV Brazilian Symposium on Integrated Circuits and Systems Design*.

[7] IEEE Standard VHDL Language Reference Manual, IEEE Std 1076-1993.

[8] IEEE Standard VHDL Analog and Mixed-Signal Extensions, IEEE Std 1076.1-1999.

[9] Alex Doboli, Ranga Vemuri, "Behavioral Modeling for High-Level Synthesis of Analog and Mixed-Signal Systems From VHDL-AMS", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1504-1520,Vol. 22, No. 11, November 2003.

[10] Ernst Christen, Kenneth Bakalar, "VHDL-AMS – A Hardware Description Language for Analog and Mixed-Signal Applications", *IEEE Transactions on Circuits and Systems – II: Analog and Digital Signal Processing*, pp. 1263-1272, Vol. 46, No. 10, October 1999.

[11] Peter J. Ashenden, Gregory D. Peterson, Darrell A. Teegarden, *The System Designer's Guide to VHDL-AMS*, Morgan Kaufmann, 2002.