10

FPGA Implementation of PID Controller for the Stabilization of a DC-DC "Buck" Converter

Eric William Zurita-Bustamante[1], Jesús Linares-Flores[2], Enrique Guzmán-Ramírez[2] and Hebertt Sira-Ramírez[2]
[1]*Universidad del Istmo*
[2]*Universidad Tecnológica de la Mixteca*
México

1. Introduction

Actually the development of control systems in embedded systems presents a great advantage in terms of easy design, immunity to analog variations, possibility and implement complex control laws and design a short time (Mingyao Ma et al., 2010). One of the devices that allows embedded systems arrangements are field-programmable gate array (FPGA). Several advantages of using FPGAs in industrial applications can be seen in (Joost & Salomon, 2005).

The use of FPGAs to implement control laws of various systems can be observed in different articles. In (Hwu, 2010) performance a technique based on a field programmable gate array to design PID controller applied to the forward converter to reduce the effect of input voltage variations on the transient load response of the output converter. The main characteristic of this technique is the on-line tuned parameters of the PID controller. To validate the topology implemented, they designed a forward converter with an input voltage of 12V, and output dc voltage of 5.12V with a rated output DC current of 10A and a switching frequency at rated load of 195 kHz. The results show than the measured transient load response has no oscillation with on-line tuning applied to the controller.

In the work of LI et al. (Bo Li et al., 2011) presents a digital pulse-width-modulator based sliding-mode controller and FPGA for boost converter. The proposed model they used was higher order delta-sigma modulator. The problem with this modulator is the stability problem. To resolve this problem they implemented a Multi-stage-noise shaping delta-sigma DPWM (MASH sigma-delta DPWM). To verify the function of the proposed controller they implemented a boost converter connected to a Virtex-II Pro XC2VP30 FPGA with and Analog to digital converter as interface. The experimental results show than the MASH sigma-delta DPWM has a faster recovery time in load changes, compared with a PID controller.

In (Mingyao Ma et al., 2010) proposed a FPGA-based mixed-signal voltage-mode controller for switching mode converters. The architecture of the scheme consists of a DPWM generation with a PID controller implemented on FPGA, a DAC and a comparator. The switching mode converters state variables are digitalized via an ADC to the PID controller. The control signal goes to the DPWM module to generate the PWM waveforms. They implemented the PID and the DPWM on a Cyclone II series EP2C25, in other hand; they implemented a single phase

full-bridge inverter like the switching mode converter to test the architecture of the controller. Their architecture allows integration of a control system in FPGA.

An implementation of PID controller on FPGA for low voltage synchronous buck converter is presented in (Chander et al., 2010). They use MATlab/Simulink for the PID controller design to generate the coefficients of the controller. They did a comparison between different coefficients to obtain a reasonable controller for the converter. The architecture was implemented in FPGA Virtex-5 XC5VLX50T.

In this article, we will focus on the PID average output feedback controller, implemented in an FPGA, to stabilize the output voltage of a "buck" power converter around a desired constant output reference voltage. The average control inputs are used as a duty ratio generator in a PWM control actuator. The architecture control, used for the classical PID control, has the following features:

- The PWM actuator is implemented through a triangular carrier signal and a comparator. The main function of this modulator is the average signal conversion to a pulsing signal that activates and deactivates the converter power transistor, at a switching frequency of 48kHz.
- The processing time control for the PID is 20.54μs. This processing time were achieved thanks to the parallel execution of units modeled within a FPGA Monmasson & Cirstea (2007)-Rogriguez-Andina et al. (2007).
- The output voltage is obtained through an Analog to Digital Converter (ADC), which is the only additional hardware needed to operate to the controllers. The used ADC is the ADC0820, which is an 8 bits converter.

The rest of the document is organized as follows: section 2 presents the mathematical model of the "buck" converter. The design of the PID control is shown in the section 3, while the simulation of the PID control design is presented in section 4. The architecture of the implemented control is found in section 5. The experimental results of the implementation of the FPGA based controller, are found in section 6. Finally, the conclusions of this work are given section 7.

2. The "buck" converter model

Consider the "buck" converter circuit, shown in Fig. 1. The system is described by the following set of differential equations:

$$\begin{aligned} L\frac{di_L}{dt} &= -v_0 + Eu \\ C\frac{dv_0}{dt} &= i_L - \frac{v_0}{R} \\ y &= v_0 \end{aligned} \tag{1}$$

where i_L represents the inductor current and v_0 is the output capacitor voltage. The control input u, representing the switch position function, takes values in the discrete set 0, 1. The system parameters are constituted by: L and C which are, respectively, the input circuit inductance and the capacitance of the output filter, while R is the load resistance. The external voltage source exhibits the constant value E. The average state model of the "buck" converter circuit, extensively used in the literature (a) Linares & Sira, 2004; b) Linares & Sira, 2004;

Linares et al., 2011; Sira & Agrawal, 2004) may be directly obtained from the original switched model, (1), by simply identifying the switch position function, u, with the average control, denoted by u_{av}. Such an average control input is frequently identified with the duty ratio function in a Pulse Width Modulation implementation. The control input u_{av} is restricted to take values in the closed interval [0, 1]. From (1), the "buck" converter system is clearly a second order linear system of the typical form: $\dot{x} = Ax + bu$ and $y = c^T x$.

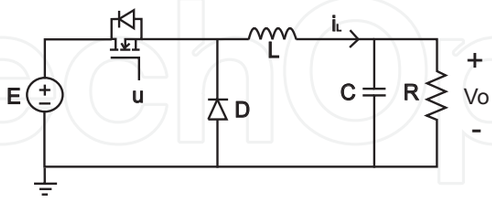

Fig. 1. The electrical circuit of the "buck" converter.

$$\begin{aligned} A &= \begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix} \\ b &= \begin{bmatrix} \frac{E}{L} \\ 0 \end{bmatrix} \\ c^T &= \begin{bmatrix} 0 & 1 \end{bmatrix} \end{aligned} \tag{2}$$

Hence, the Kalman controllability matrix of the system $\mathcal{C} = [b, Ab]$, is given by:

$$\mathcal{C} = \begin{bmatrix} \frac{E}{L} & 0 \\ 0 & \frac{E}{LC} \end{bmatrix} \tag{3}$$

The determinant of the controllability matrix is ($\frac{E^2}{L^2C} \neq 0$). Therefore, the system is controllable (Dorf & Bishop, 2011), now we design a classic PID control in the following section.

3. PID controller design

The FPGA implementation of a classical Proportional Integral Derivative (PID) controller was designed based on the corresponding transfer function of the converter (Ogata, 2010), obtained from the average model given in (1), is

$$\frac{V_0(s)}{U_{av}(s)} = \frac{\frac{E}{LC}}{s^2 + \frac{1}{RC}s + \frac{1}{LC}} \tag{4}$$

While the transfer function of the PID controller, is:

$$F_{PID}(s) = K_p(1 + \frac{1}{T_i s} + T_d s) \tag{5}$$

The block diagram of the PID controlled system is shown in Fig. 2.

The closed loop transfer function is readily found to be

$$H(s) = \frac{(K_p T_d T_i s^2 + K_p T_i s + K_p)(\frac{E}{LC})}{s^3 + (\frac{1}{RC} + \frac{EK_p T_d}{LC})s^2 + \frac{(1+EK_p)}{LC}s + \frac{EK_p}{LCT_i}} \tag{6}$$

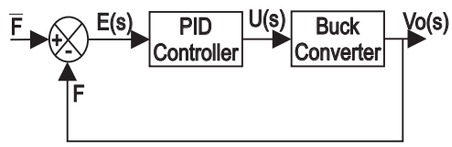

Fig. 2. PID control in closed loop.

The closed loop characteristic polynomial of the PID controlled system is then given by

$$s^3 + (\frac{1}{RC} + \frac{EK_pT_d}{LC})s^2 + \frac{(1+EK_p)}{LC}s + \frac{EK_p}{LCT_i} = 0 \tag{7}$$

The coefficients K_p, T_i and T_d are chosen so that (7) becomes a third order Hurwitz polynomial of the form (Dorf & Bishop, 2011; Ogata, 2010):

$$p(s) = (s^2 + 2\zeta\omega_n s + {\omega_n}^2)(s+\alpha) \tag{8}$$

Equating the characteristic polynomial coefficients (7) with those of the desired Hurwitz polynomial (8), we obtain the following values of the parameters for the PID controller,

$$\begin{aligned} K_p &= \frac{2\zeta\omega_n\alpha LC + {\omega_n}^2 LC - 1}{E} \\ T_i &= \frac{EK_p}{LC\alpha{\omega_n}^2} \\ T_d &= \frac{LC}{EK_p}(\alpha + 2\zeta\omega_n - \frac{1}{RC}) \end{aligned} \tag{9}$$

4. PID controller cosimulation

In this section, we develop the simulation of the PID controller. This simulation is performed using Matlab/Simulink, ModelSim and PSim Software.

The cosimulation in Matlab/Simulink creates an interface between Matlab and Matlab external program, i.e., the cosimulation allows the interaction of an external simulator with Matlab tools. The cosimulation provides a fast bidirectional link between the hardware description language (HDL) simulator, and Matlab/Simulink for direct hardware design verification Matlab (2008).

Figure 3 shows the scenario between ModelSim and Matlab to obtain the cosimulation. The block that allows interaction with the HDL simulator is called "*EDA Simulator Link MQ*".

The PSIM software includes an application called *SimCoupler* that presents an interface between PSIM and Matlab Simulink for cosimulation. With the module *SimCoupler* part of the system can be implemented and simulated in PSIM, and the rest of the system in Simulink. With this tool we can access to the broad features of Psim simulation, and the capabilities of Simulink simulation in a complementary way Psim (2006).

The module *SimCoupler* consist of two parts: the link nodes in PSim, and the SimCoupler model block in Simulink. In this work we use the module *SimCoupler* to simulate the buck converter in Psim, while in matlab and through another cosimulation part of the PID control. Figure 4 shows the buck converter circuit in Psim, in this figure one can observe that the circuit

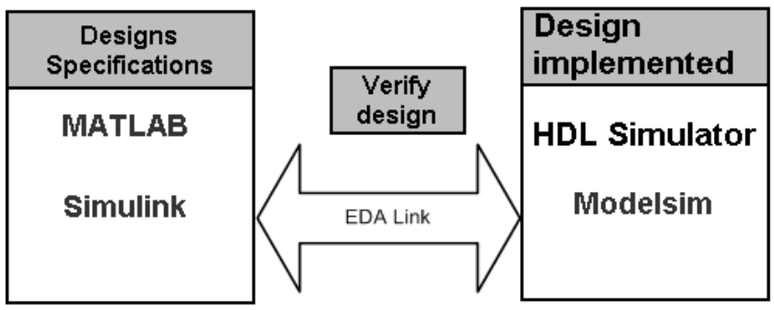

Fig. 3. Cosimulation between Matlab and ModelSim.

input is the signal coming from the PWM control in Simulink, this input signal is connected using the *In link node*. Because the system is feedback, and the feedback signal is the output voltage of the buck converter (V_o), we use the *Out link node* to send the output voltage to Simulink.

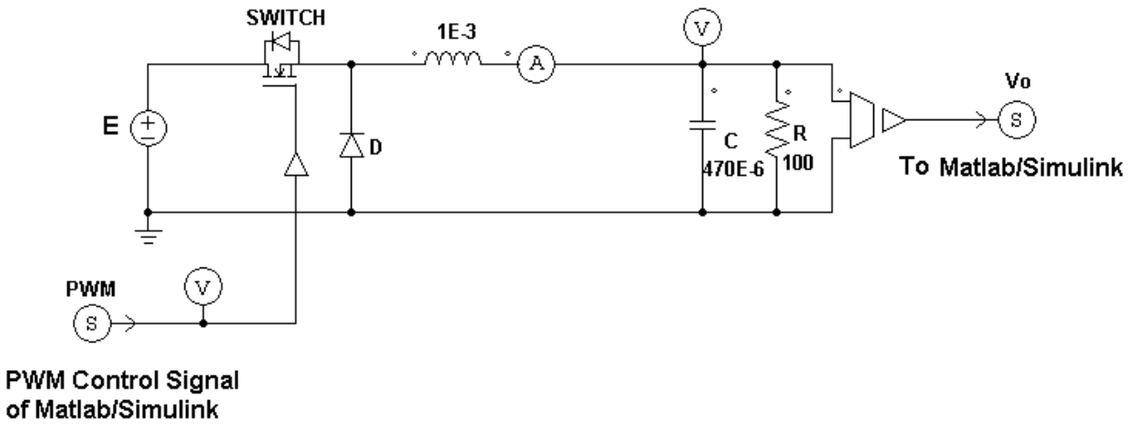

Fig. 4. Buck converter circuit for cosimulation with Simulink.

In Simulink we choose the S-function SimCoupler library and the SimCoupler block are added to the design. After adding the block, in its properties is chosen the file path for cosimulation in Psim, a window will automatically appear as shown in Fig. 5 showing all inputs and outputs of the circuit in PSIM, in this case, according to the diagram there are one input and one output on the circuit, the output voltage of the buck converter V_o. While that the input is the PWM signal.

Once set the block are automatically displayed input and output signals in the block for subsequent cosimulation, as shown in Fig. 6.

Before simulating the final system, we proceed to simulate the performance of open-loop buck converter, for this, we just simulate the circuit in Psim. Figure 7 shows the output voltage in simulation for open-loop. The response presents a overshoot of 100% however are able to stabilize around 45ms.

On the other hand, we simulate the PID control with the tools of differentiation and integration of Simulink. Figure 8 shows the performance of the PID controller with

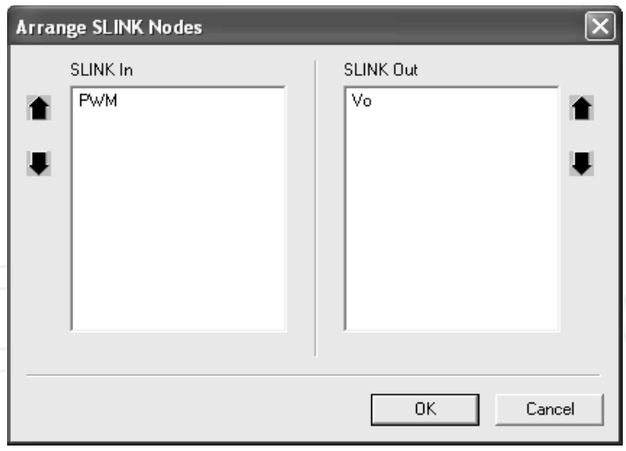

Fig. 5. Inputs and Outputs of the PSIM circuit.

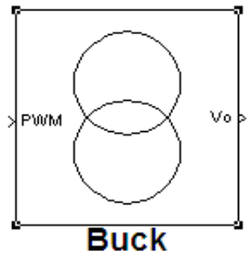

Fig. 6. Simulink block with the inputs and outputs of the PSIM circuit.

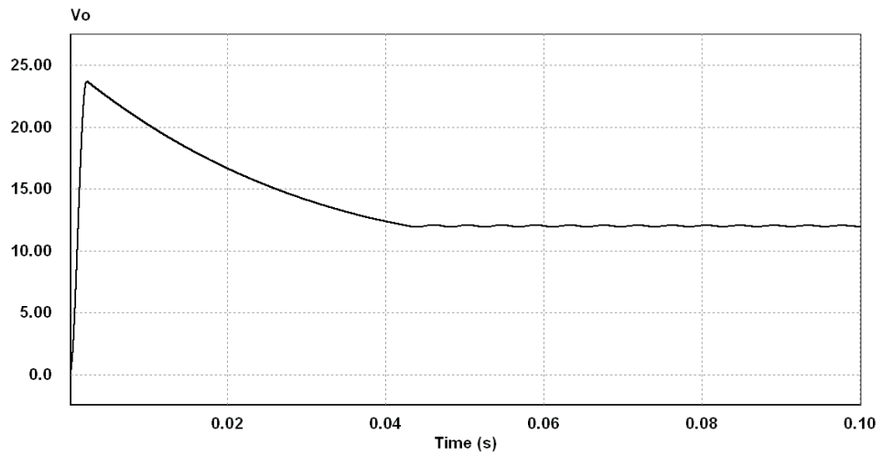

Fig. 7. Output voltage of buck converter in open-loop.

cosimulation between Matlab/Simulink and Psim. The PID control stabilizes the output voltage signal in a time of approximately 18ms, greatly decreasing the overshoot presented at the open-loop response.

Figure 9 shows a cosimulation for the final system with a desired output voltage of 4V, and shows that in the transient response has not overshoot, however, the settling time is about 23 ms, what is intended to improve with the experimental results. Also the Fig. 10 shows the output voltage for a desired voltage of 18 V, which shows that it has a maximum overshoot

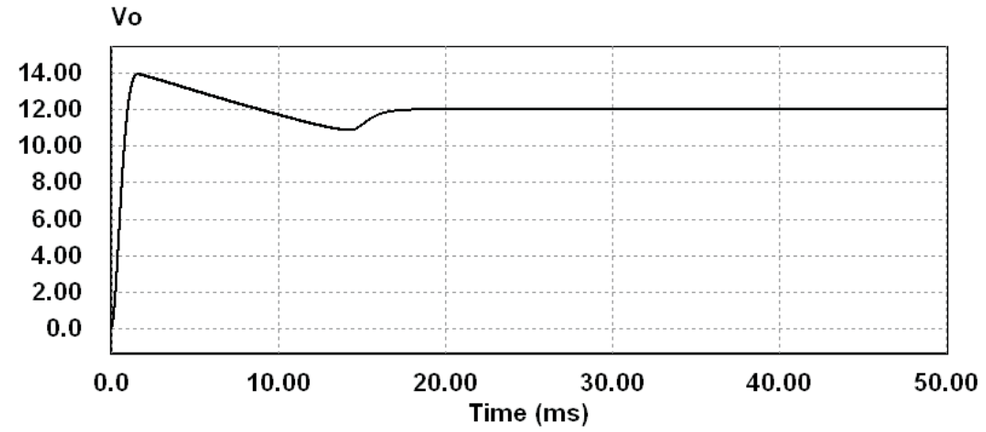

Fig. 8. Output voltage of buck converter with PID controller in cosimulation.

of 7.6 %, and a maximum error of 0.15 V. According to these simulations, we proceed to implement the system on a FPGA NEXYS2 board.

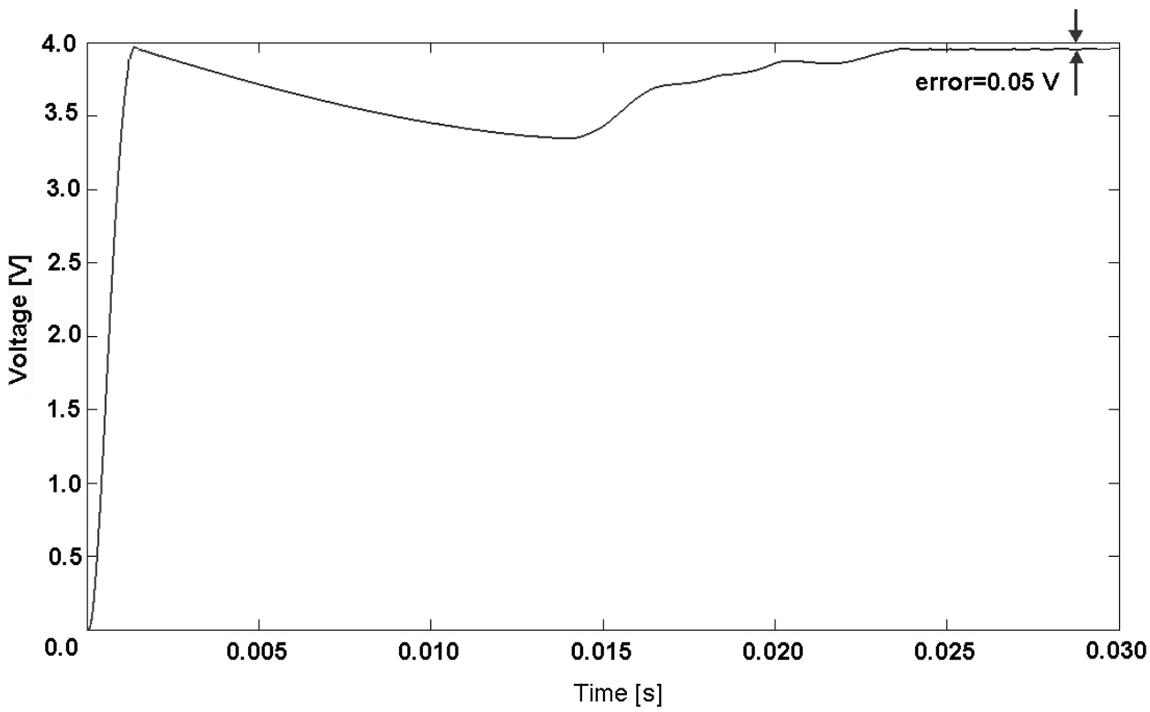

Fig. 9. Output voltage of buck converted with a desired voltage of 4 V in cosimulation.

5. Discrete PID controller implemented on the FPGA

In this section, we explain the hardware implementation of the discrete PID controller. For this purpose, we used the Xilinx ISE Design Suite 12.2 EDA (electronic design automation) -software tool and the Spartan 3E board EDA-hardware tool, it includes a Xilinx Spartan-3E1600 FPGA.

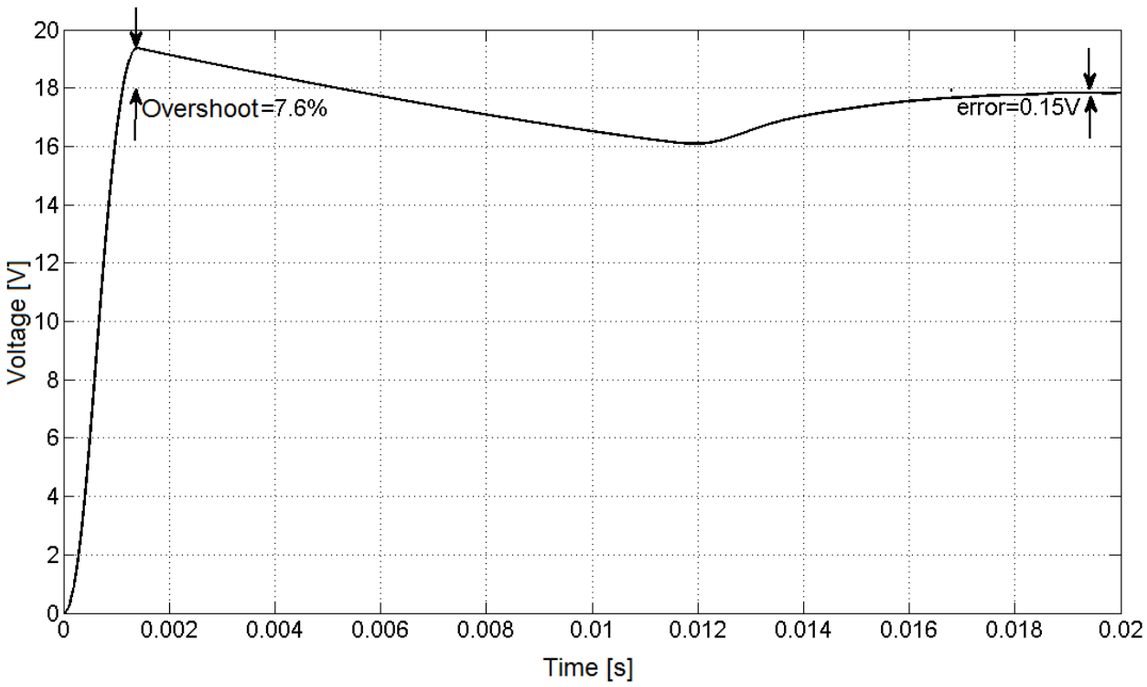

Fig. 10. Output voltage of buck converted with a desired voltage of 4 V in cosimulation.

Now, we must define an efficient design methodology and the abstraction level to model the system, and choose an appropriate sampling period and the suitable format for coefficients and variables.

The PID controller design is based on a hierarchical and modular approach using Top-Down methodology (Palnitkar, 2003), where the modules can be defined with diverse levels of abstraction. Thus, for this design the schematic description was chosen as top level and the controller components were modeled with the VHDL hardware description language (using a behavior level modeling). Previous analysis and simulations showed that due to the range of results generated by the operations involved in the discrete controller is necessary to use a floating point format; for this intention, the IEEE Standard for Binary Floating-Point Arithmetic, IEEE Std 754-1985 (IEEE, 1985) was chosen. Now, based on top-down methodology, an initial modular partitioning step is applied on the FPGA-based PID controller, this process generate four components, Clock manager, ADC control, Control law and PWM generator (see Fig. 11).

The PID controller work with a frequency of 50 MHz (Clk_PID). The Clk_main signal is generated from Clk_main signal by the Clock manager component. The principal element of this component is the Digital Clock Manager (DCM). The DCM is embedded on the Spartan3E FPGA's families and it provides flexible complete control over clock frequency, maintaining its characteristics with a high degree of precision despite normal variations in operating temperature and voltage. The DCM provides a correction clock feature, ensuring a clean Clk_PID output clock with a 50% duty cycle.

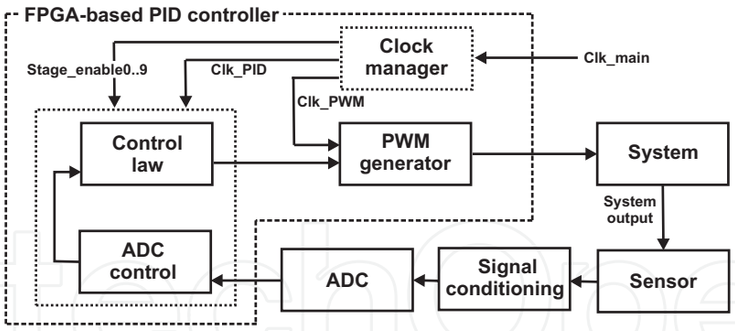

Fig. 11. Block diagram of FPGA-based PID controller.

In order to increase the performance of the control system, we proposed pipeline architecture for the PID controller. Therefore, enable signals of pipeline registers (Stage_enable0..9) are required. These signals are also generated by the Clock manager component.

In addition, the clock manager component generates the frequency required by the PWM for its operation (Clk_PWM). The Clk_PWM signal is derived from the Clk_main by a Digital Frequency Synthesizer (DFS) included in the DCM. The frequency of the Clk_PWM is 25 MHz and has a 50% duty cycle correction too.

The Information from the sensor is analog source, so it must be discretized for that the FPGA can process. For this purpose we have chosen the Analog-Digital Converter (ADC) ADC0820. The ADC0820 is an 8 bits resolution converter, it offers a 2μs conversion time and it has a 0 to 5 Volts analog input voltage range. The element responsible for this task is the ADC control component.

The ADC control component is composed of two modules, the ADC interface module, which is a simple finite-state machine (FSM) that implements the communications protocol to acquire data of the ADC0820, and the float-point encoder module, which converts the integer value into single-precision floating-point format. A block diagram of ADC interface module is shown in Fig. 12.

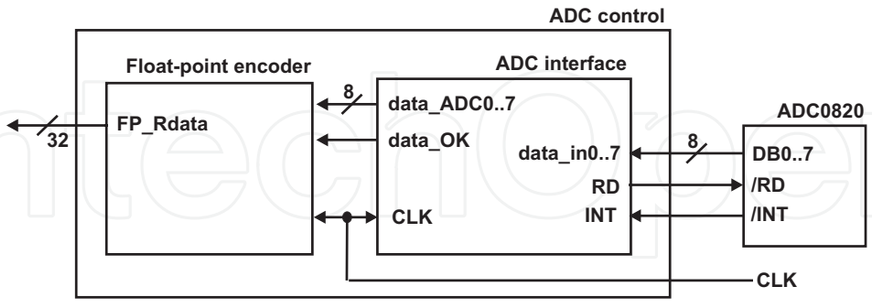

Fig. 12. Block Diagram of the ADC control component.

Now, the information generated by the ADC control component should be processed by the corresponding control law.

The discrete PID controller was synthesized on a FPGA based on equations for the continuous PID controller (Ogata, 2010), defined as

$$u_{av} = K_p(\overline{F}(t) - F(t)) + K_i \int_0^t (\overline{F}(t) - F(t))dt + K_d \frac{d(\overline{F}(t) - F(t))}{dt} \tag{10}$$

where $K_i = \frac{K_p}{T_i}$ and $K_d = K_p T_d$.

An important aspect in the discretization of (10) is the obtaining of a discrete approximation of the continuous integral and a discrete approximation of the continuous derivative.

For discrete approximation of the continuous integral we have used the Adams-Bashforth method of the second order (Ascher & Petzold, 1998). This method is defined as

$$y[n+1] = y[n] + \frac{1}{2}\Delta t(3\dot{y}[n] - \dot{y}[n-1]) \tag{11}$$

Then, if the continuous integral is defined as $\int_0^t (\overline{F}(t) - F(t))dt$, using the Adams-Bashforth method, its discrete approximation is defined as

$$F[n+1] = F[n] + \frac{1}{2}\Delta t(3(\overline{F}[n] - F[n]) - (\overline{F}[n-1] - F[n-1])) \tag{12}$$

The Fig. 13 shows the proposed architecture for discrete approximation of a continuous integral given by (12).

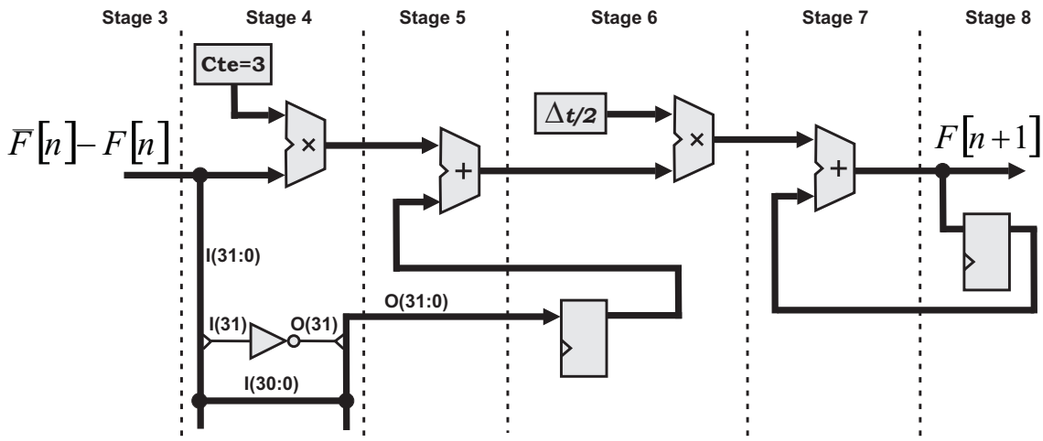

Fig. 13. Block diagram of the discrete approximation of a continuous integral.

On the other hand, the discrete approximation of the continuous derivative is obtained based on finite differences method of the first order (Burden & Douglas, 2000), using the backward difference. This method is defined as

$$(\frac{\partial y}{\partial t})_n \approx \frac{y[n] - y[n-1]}{\Delta t} \tag{13}$$

Then, if the continuous derivative is defined as $\frac{\partial(\overline{F}(t) - F(t))}{\partial t}$, using the finite differences method, its discrete approximation is defined as

$$F'[n] = \frac{(\overline{F}[n] - F[n]) - (\overline{F}[n-1] - F[n-1])}{\Delta t} \tag{14}$$

The Fig. 14 shows the proposed architecture for discrete approximation of a continuous derivative given by (14).

The architecture consists of six multipliers and six adders. Then, it is necessary to implement single-precision floating point custom-adder and custom-multiplier.

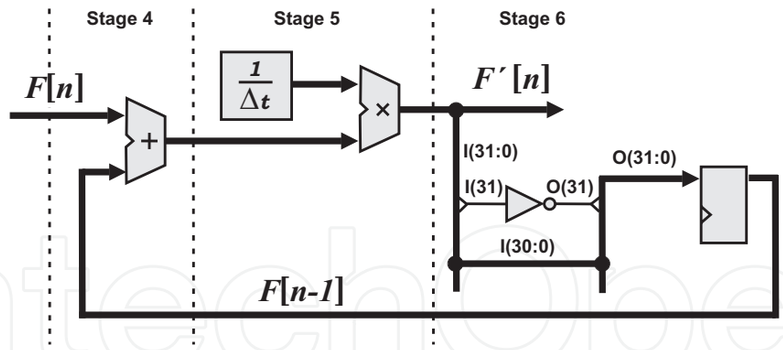

Fig. 14. Block diagram of the discrete approximation of a continuous derivative.

The Xilinx ISE Design Suite 12.2 includes the CORE Generator tool, which allows generating pre-optimized elements for Xilinx's FPGA. Our controller architecture uses multipliers and adders of single-precision floating-point, standard Std-754, generated by this tool. The symbols of the multiplier and adder generated by the CORE Generator tool are showed in the Fig. 15.

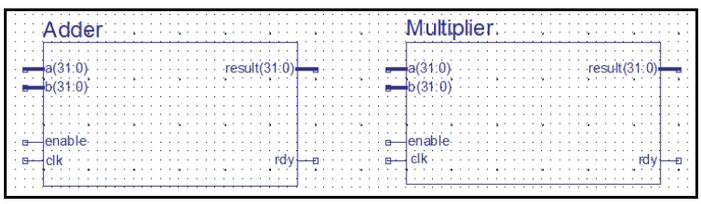

Fig. 15. Adder and Multiplier modules generated for Xilinx CORE Generator tool.

The proposed PID controller architecture is composed of 10 pipeline stages (see Fig. 16) and each of them needs 100 cycles to fulfill its function (2 μs), this indicates that the processing time of one data is 20 μs (time between 2 consecutive data delivered by the controller to the next component, the PWM). The enable signals (Stage_enable0..9) have the control each one of the pipeline registers that composed the proposed architecture.

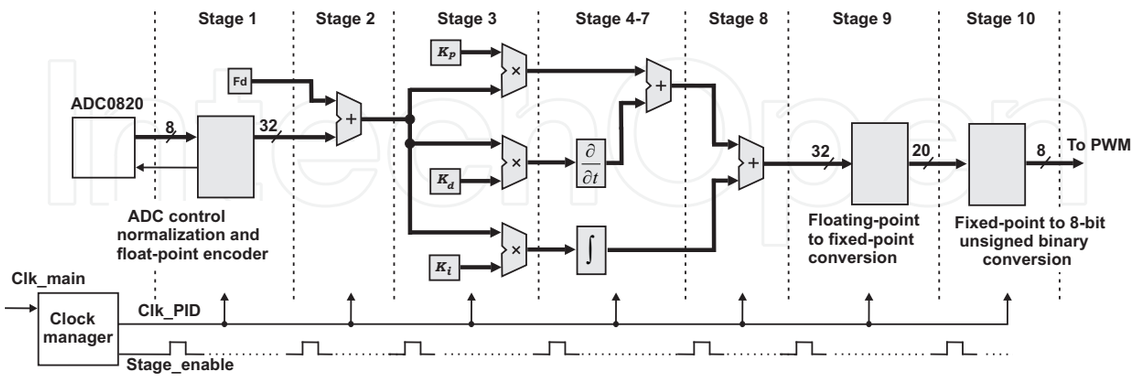

Fig. 16. Architecture proposed for the discrete PID controller implemented into the Spartan-3E1600 FPGA.

In the last stage the PID controller output must be adequacy for the PWM module. This adequation consists of Float-point to 8 bit unsigned binary conversion.

The last component of the proposed architecture is the PWM. The PWM component consists of single up-down counter unit and one magnitude comparator unit (see Fig. 17(a)).

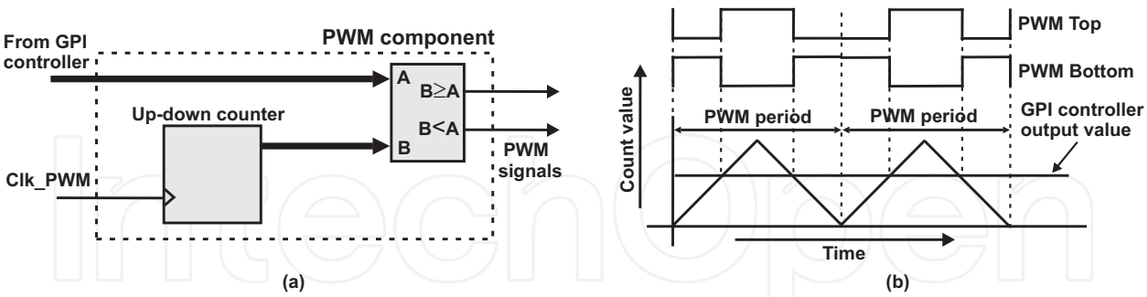

Fig. 17. (a) PWM component; (b) PWM outputs.

The PWM output frequency depends of the maximum count value of the counter and the Clk_PWM frequency (see figure 17(b)). Then, the PWM output frequency is defined as

$$PWMfrequency = \frac{CLK\_PWM}{2(maximuncount + 1)} = \frac{25MHz}{512} = 48.828KHz \tag{15}$$

The implementation result of the complete architecture for discrete PID controller are reported in Table 1.

Mod.	Slices	Flip-Flops	4-input's -LUT's	Pre-opt -elem.	Max. Freq (MHz)
1	5668 (38%)	8722 (29%)	8737 (29%)	1 BRAM (2 %) 1 DCM (12 %)	60.37

Table 1. Discrete PID controller implementation results.

6. Experimental results

The PID control and the Pulse Width Modulator (PWM) actuator for the regulation of output voltage of the buck converter were implemented in a Spartan 3E board. The only external hardware connected to the FPGA for measuring the "buck" converter output voltage was the analog digital converter ADC0820. Figure 18 illustrates the block diagram of the FPGA-based control system based on PID controller.

6.1 Requirements of the PID controller

Figure 19 shows the open-loop response of the "buck" converter with the following specifications: $L = 1mH$, $C = 100\mu F$, $R = 100\Omega$, $E = 24V$, $f = 48.828KHz$, $\Delta v_0/v_0 = 0.013\%$, $\Delta i_L = 0.092$ and a duty cycle $D = 0.75$. The output voltage response is a steady-state error of 5.56% and has a settling time of 15ms. On the other hand, we get that the diagram bode of the transfer function given by (4) with the same parameters, has a gain margin $Gm = Inf$(at Inf rad/sec) and a phase margin $Pm = 0.377$deg (at 1.58×10^4 rad/sec). Given that the buck converter system has infinite gain margin, it can withstand greater changes in system parameters before becoming unstable in closed loop. Since the system has this characteristic, we will design our controllers in closed loop with the following requirements: Overshoot

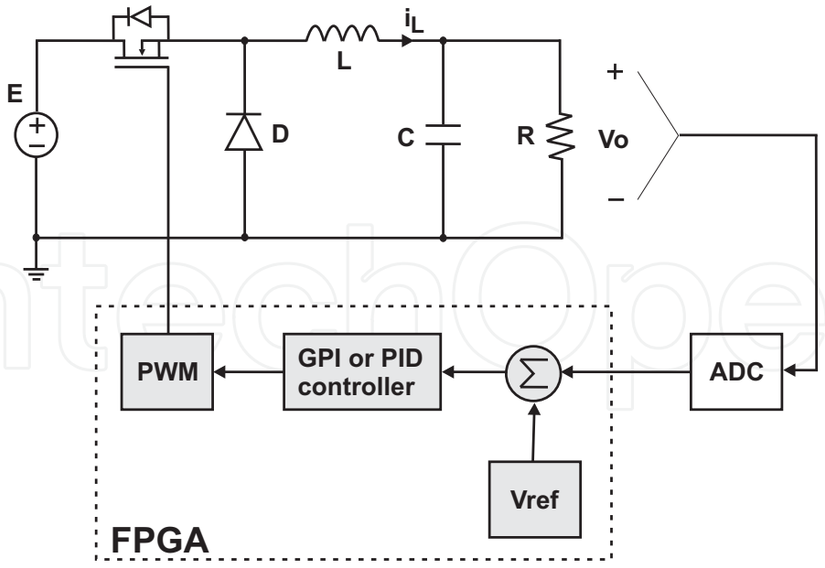

Fig. 18. Block diagram of the FPGA-based control system for PID controller.

less than 4.32%, Setting time less than 5 milliseconds, Steady-state error less than 1%, and Maximum sampling time $40\mu s$.

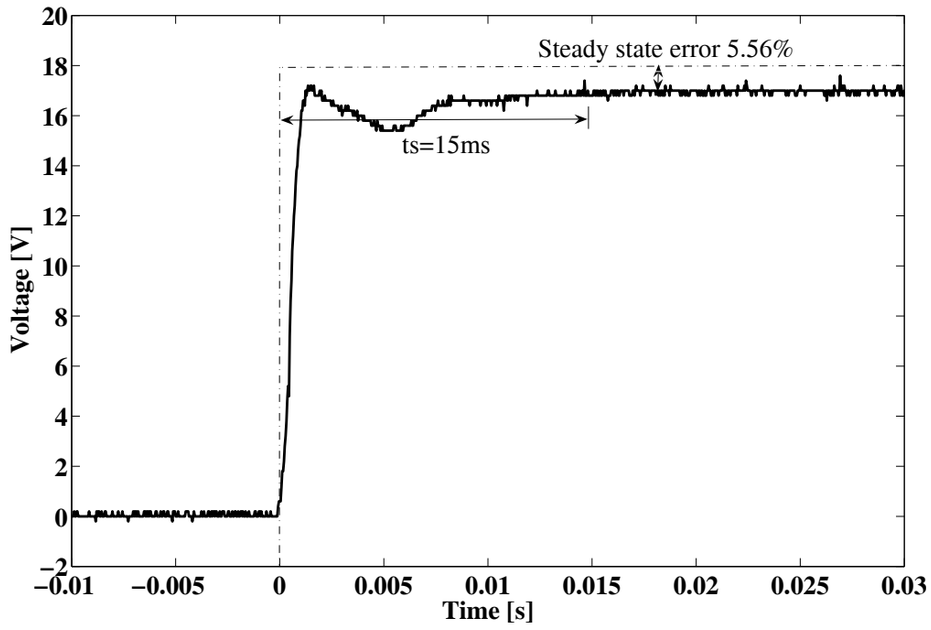

Fig. 19. Output voltage transient response of the "buck" converter with the PID control scheme.

The PID controller gains obtained by the design requirements were:

$$K_p = 0.15; \quad T_i = 1.2 \times 10^{-3}; \quad T_d = 5.9 \times 10^{-4} \tag{16}$$

6.2 PID controller into the FPGA

Figure 20 shows the performance of the PID control law, in the stabilization task for the "buck" converter output voltage. As before, we used a constant reference of 18 V. The continuous line corresponds to the PID controlled response. The settling time of the response of the "buck" converter output voltage through the PID controller, is 13.64 ms. The PID controller tuning was done through a third order Hurwitz polynomial.

Table 2 exhibits the performance of the synthesized controller. The main specifications of the transient response, the bandwidth of the PID controller (see Table 2), these frequencies are calculated in the closed-loop through the damping ratio and settling time (Messner & Tilbury, 1999). The damping coefficient value is 0.707, while the value of settling time is: 13.64 ms.

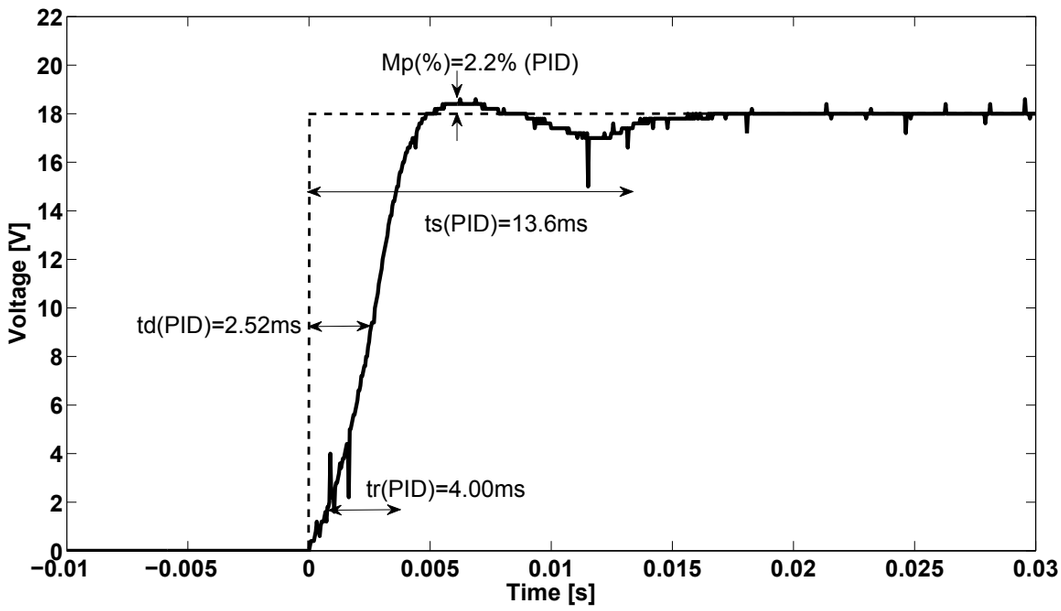

Fig. 20. Output voltage transient response of the "buck" converter with the PID control scheme.

Delay time	t_d	2.52 ms
Rise time	t_r	4 ms
Time of peak	t_p	6.24 ms
Percentage of overshoot	M_p	2.2 %
Settling time	t_s	13.64 ms
Bandwith	B_ω	414.85 Hz

Table 2. Specifications of the Controller transient response PID.

To illustrate the robustness of the PID controller, we made a test with the "buck" converter system by suddenly connecting a dynamic load (DC motor) at the output of the "buck" converter. Figure 21(a) shows the behavior of the perturbed converter's output voltage and the recovery of the output voltage to the desired reference signal when the converter is controlled with the PID controller scheme. Also, in Figure 21(b) is shown the u_{av} control signal, from the PID scheme implemented in the FPGA.

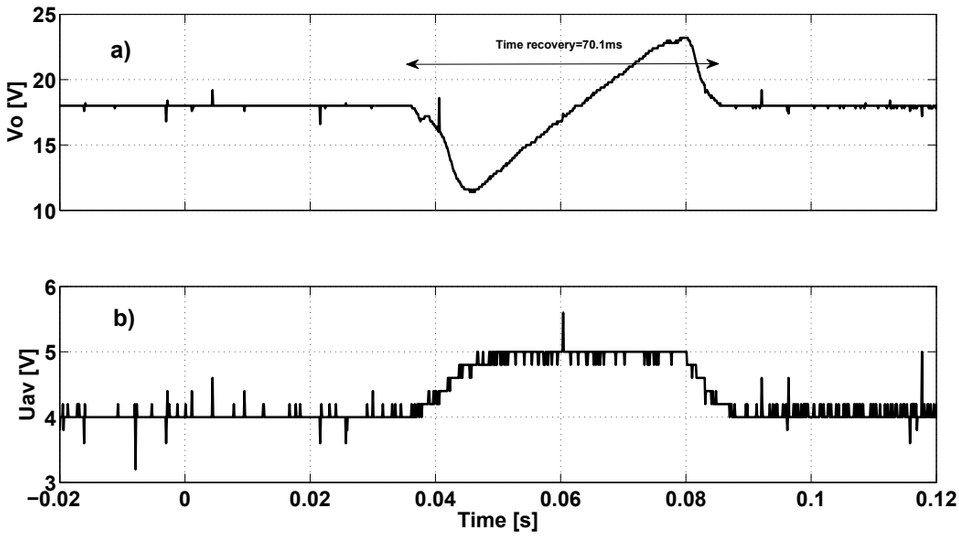

Fig. 21. Output voltage response of the "buck" converter with sudden connection of a DC motor.

7. Conclusions

In this work, we have applied the Proportional Integral Derivative control scheme, synthesized via a Field Programmable Gate Array implementation, for the output voltage regulation in a DC/DC power converter of the "buck" type. The performance of the PID control action was synthesized via a FPGA. The results obtained by cosimulation allowed to study each of the units designed and modeled in VHDL, correcting some errors and, in addition, the cosimulation was a perfect tool allowing faster design process to get a full system simulation before implement the system in the FPGA board. Also we conclude that the PID controller has a good transient response. When we connect a static and a dynamic load to the "buck" converter output, we observed that the PID control results in a significantly faster response, regarding the output voltage recovery time to the desired reference. Finally, the experimental results show the effectiveness of the FPGA realization of both the PID controller, in this case, programmed into the FPGA. This methodology of design can be used to design switched mode power supplies with efficiency greater than 95%.

8. References

Ascher, U. & Petzold, L. (1998). Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations. SIAM: Society for Industrial and Applied Mathematics.

Bo Li; Shuibao Guo; Xuefang Lin-Shi & Allard, B. (2011) , Design and implementation of the digital controller for boost converter based on FPGA,*IEEE International Symposium on Industrial Electronics (ISIE)*, pp. 1549-1554.

Burden, R. & Douglas, F. (2000). Numerical Analysis. Brooks/Cuelo.

Chander, S.; Agarwal, P. & Gupta, I. (2010) , FPGA-based PID controller for DC-DC converter, *Power Electronics, Drives and Energy Systems (PEDES) & 2010 Power India, 2010 Joint International Conference on*, Dec. 2010, pp.1-6.

Dorf, R. C. & Bishop,R. H., (2011).*Modern Control Systems*.Twelfth Edition: Prentice-Hall.

Hwu, K. I. (2010). Forward Converter with FPGA-Based Selft-Tuning PID Controller. *Tamkang Journal of Science and Engineering*, Vol. 13, No. 2, June 2010, 173-180, ISSN:1560-6686

IEEE Computer Society. (1985). IEEE Standard for Binary Floating-Point Arithmetic, IEEE Std 754-1985.

Joost, R. & Salomon, R., (2005). Advantages of FPGA-based multiprocessor systems in industrial applications, *Industrial Electronics Society, 2005. IECON 2005. 31st Annual Conference of IEEE*, Nov. 2005, pp. 445-450.

Linares-Flores, J. & Sira-Ramírez, H. (2004). DC motor velocity control through a DC-to-DC power converter, *Proc. 43rd IEEE Conf. Dec. Control*, pp. 5297-5302.

Linares-Flores, J. & Sira-Ramírez, H. (2004). Sliding Mode-Delta Modulation GPI Control of a DC Motor through a Buck Converter,*Proc. 2nd Symposium on System, Structure and Control*, México.

Linares-Flores, J.; Antonio-García, A. & Orantes-Molina, A. (2011). Smooth Starter for a DC Machine through a DC-to-DC buck converter, *Revista Ingeniería, Investigación y Tecnología*, Universidad Autónoma de México (UNAM), Vol. XII, No. 2, April 2011, ISSN: 1405-7743.

The MathWorks, Inc. (2008). Release Notes for Matlab R2008a.

Messner, W. C. & Tilbury, D. W., (1999). *Control tutorial for matlab and simulink: A web-Based Approach*, Addison-Wesley.

Mingyao Ma; Wuhua Li & Xiangning He. (2010), An FPGA-based mixed-signal controller for switching mode converters,*IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, Nov. 2010, pp.1703-1708.

Monmasson, E. & Cirstea, M. N., (2007). FPGA design methodology for industrial control systems-A review, *IEEE Transactions on Industrial Electronics*, vol. 54, no. 4, August 2007, pp. 1824-1842.

Ogata, K. (2010). *Modern Control Engineering*. 5th Edition: Prentice-Hall.

Palnitkar, S. (2003). A guide to digital design and synthesis. 2nd Edition, USA: Printece-Hall.

Phillips, C. L. & Nagle, H. T., (1995). *Digital Control Systems Analysis and Design*, Third Edition: Prentice-Hall.

Powersim, Inc. (2006). *PSIM User's Guide*

Rodriguez-Andina,J. J.; Moure, M. J. & Valdes, M. D., (2007). Features, design tools, and application domains of FPGAs, *IEEE Transactions on Industrial Electronics*, vol. 54, no. 4, August 2007, pp. 1810-1823.

Sira-Ramírez, H. & Agrawal, S. K. (2004).*Differentially Flat Systems*,Marcel Dekker, New York.

IntechOpen