DEVELOPMENT OF AN EMBEDDED CONTROLLER FOR SMALL SCALE DC MOTORS USING FPGA

V. GEETHA PRIYA[1] **P. LAKSHMI[2]** **V. ARTHI[3]**
Associate Professor **Professor** **Systems Engineer**

[1]Rajalakshmi Engineering College, Chennai, [2]CEG, Anna University, Chennai [3]TCS, Chennai
geethapriya.v@rajalakshmi.edu.in, p_lakshmi@annauniv.edu, arthi92.eee@gmail.com

Abstract - In this paper, a comprehensive method for the development of an embedded PID controller for the speed control of small scale dc motors using a low cost FPGA(Field Programmable Gate Array) is presented. The PID controller in FPGA is represented in terms of VHDL code. For choosing the controller parameters and system identification, MATLAB is used with the open loop step response of the motor obtained from an experimental setup. UART communication interface along with a glitch filter is also implemented with the controller on the same FPGA which makes it an efficient plug and play device. The simulation results for the PID controller are shown using Xilinx ISim. The hardware implementation is made in Xilinx Spartan 6 FPGA chip with a sampling time of 10ms.

Index Terms – PID controller, VHDL, FPGA, embedded control, small-scale dc motors.

I. INTRODUCTION

Small scale wheeled mobile robots(WMRs) are gaining popularity these days, in applications such as urban agriculture, rescue operations [1], defence, and surveillance. In spite of the smallness in size of the robots, to achieve the same functionalities and processing capabilities, embedded systems are designed and adopted in such robot applications. Embedded systems take real world input and provide processed output, aiming at the optimal use of memory, power and time. In a wide variety of low power applications, which are run from a dc source, PMDC motors are the most cost-effective choice [2] and they are generally used for the mobility control of Small scale robots. For the speed control of these DC motors, PID controllers are most commonly used in many industrial applications. Over the years, they are implemented in a variety of ways in analog, digital and microprocessor based circuits. Digital circuit implementation has the advantage of flexibility, precision and accuracy, greater Signal to Noise Ratio (SNR) over analog circuits. Microprocessor based implementation [3, 4] deals with prebuilt digital circuits executing the instructions, which provides greater flexibility. But the instructions are executed sequentially and therefore, parallelism obtained by the method of circuit implementation is lost. It is known that flexibility and parallelism can be obtained through Field Programmable Gate Arrays (FPGAs). This advantage was well utilised in a wide variety of applications, like CNC machines[5], temperature controllers[6], power electronic circuits[7,8] and cyclotron heating[9].

FPGA being a reconfigurable device, it provides improved performance than pure software implementation on general purpose processors and more flexibility than an ASIC solution[10]. Consequently, FPGA has the ability to act as 'blank hardware' for the end user. Earlier, because of its cost, the use of FPGA was somewhat limited. As smaller scale FPGAs with increased density are being designed in recent years, a wide variety of controllers namely, PID[11], Fuzzy[12], NN controller[13], Motion controllers[14], Control System Processor[15] have been developed by researchers. Efficiency in terms of space utilisation, power consumption and execution time have been studied extensively. However, cost effective implementation setup for small scale motors has not been addressed so far.

So, in this paper, a comprehensive method of implementing PID controller for speed control of small scale motors on FPGA has been developed. A glitch filter and UART Communication interface have also been implemented in FPGA which make the system cost effective leading to a simple plug-and-play type of set up for easy configuration with PC , whenever controller reconfiguration is needed.

The presentation of the paper is further organized as follows: section II gives the block diagram of the proposed digital speed control system, section III describes the controller implementation methodology, section IV deals with system identification, section V shows the simulation of the PID controller, section VI presents and discusses the results obtained with the complete hardware unit and section VII lists the conclusions arrived at based on the theoretical and experimental works carried out.

II. BLOCK DIAGRAM OF DC MOTOR SPEED CONTROL SYSTEM

The general block diagrams of the existing and the proposed dc motor speed control systems are shown in Fig.1. The major improvement in the proposed system is that it does

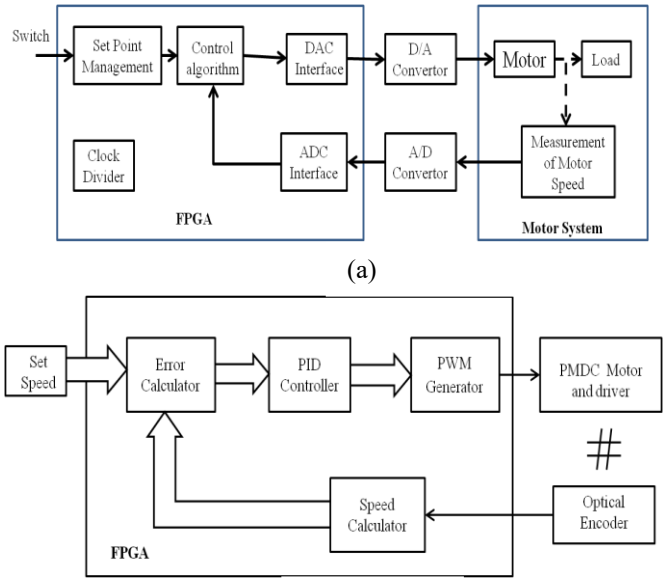

Fig. 1 Block diagram of digital speed control system (a) conventional [16] (b) Proposed

not need the A/D, D/A converters and their interfaces. Instead of A/D, a speed calculator is incorporated along with the controller itself. This calculator counts the pulses received from the encoder (optical) and computes the speed of the motor. The function of D/A converter and the interfaces in the earlier versions is carried out by a PWM generator, which is also implemented on the FPGA. So all the required hardware components are made on the available FPGA itself resulting in reduction in area of the controller which is very much expected in any well designed embedded device. PWM generator gives out pulse width modulated DC output voltage to the power electronic switch present in the driver circuit of the motor. The rotor speed of the motor is measured by counting the number of pulses obtained from the optical encoder, after removing the glitches from the pulse signal using a signal conditioner. An error calculator compares the measured speed and the set speed and its output is fed to the PID controller.

III. IMPLEMENTATION METHODOLOGY

The working sequence of the controller is given in the flowchart of Fig. 2 shows that the control system is implemented as hardware module in FPGA, where the controller components namely, P,I,D are computed simultaneously, resulting in the reduction of computation time. An optical encoder receives pulses from the motor which after processing are given as PWM pulses for motor actuation at the desired speed.

The speed of the motor can be calculated from the encoder using the equation (1), where, F is the operating frequency of FPGA device and N is the number of clock pulses received during the fixed slot interval of the encoder.

$$Speed = \frac{F*60}{N} \quad \text{.....(1)}$$

Astro, has derived an expression for the control variable, in discrete form which includes addition of three components namely, proportional, integral, derivative of error in speed as shown in equation (2) [18]. This equation has been incorporated for developing an algorithm in FPGA for control of the dc motor.

$$u(t_k) = P(t_k) + I(t_k) + D(t_k) \quad \text{.....(2)}$$

Each of these components are described using the eqns.(3-5). Since each of these components are independent of each other they lend for parallel implementation [11] so that the controller output is obtained before the next sample of error in speed is received. Error in speed is calculated by the error calculator module as a difference of set speed and actual speed. Since these modules depend on the other output, it is implemented as sequential modules.

$$P(t_k) = K\ (y_{sp}(t_k) - y(t_k)) \quad \text{.....(3)}$$

$$I(t_k) = I(t_{k-1}) + \frac{K\ h}{T_i} e(t_k) \quad \text{.....(4)}$$

$$D(t_k) = \frac{T_d}{T_d + Nh} D(t_{k-1}) - \frac{KT_d N}{T_d + Nh} (y(t_k) - y(t_{k-1})) \quad \text{..(5)}$$

The output of the controller is used to generate PWM pulses by the PWM generator module where the ON-period is given by the equation (6). [18]

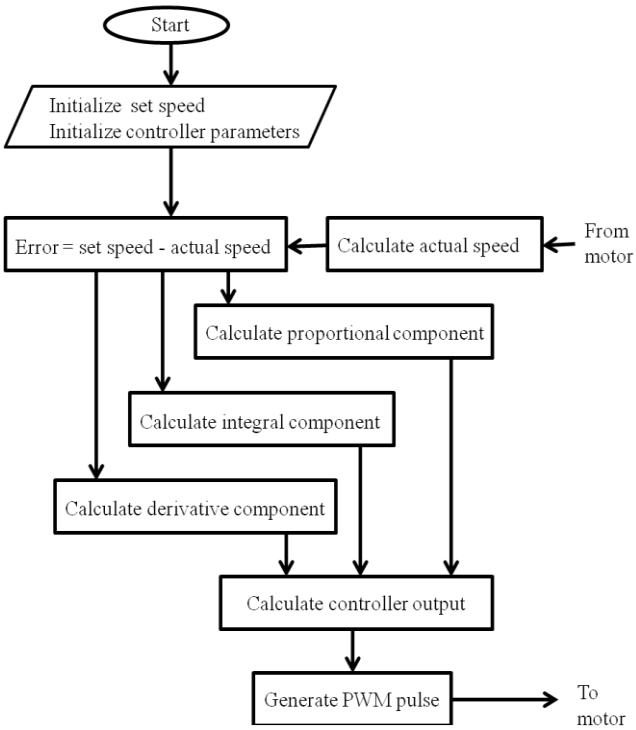

Fig. 2 Flowchart for the implementation of control system

.

$$T_{pulse}(t) = \frac{u(t) - u_{\min}}{u_{\max} - u_{\min}} T_{cycle} \quad \text{.....(6)}$$

IV. SYSTEM IDENTIFICATION

The experimental setup built for the speed control of a permanent magnet DC motor rated for 24V, 200RPM, 300 mA, is shown in Fig. 3.

Firstly the system transfer function is identified by its open loop response for a set speed of 200RPM. Since the operation reached is without the closed loop controller, final speed reached is only 175 RPM as shown in Fig. 4. However, this response is made use of along with MATLAB system identification toolbox, to obtain an appropriate transfer function that would represent the mathematical model of the system. The transfer function thus obtained is given by equation (7). This is further validated using the simulation of open loop response in MATLAB and the results are shown using the graph in Fig. 4. Equation (7) was then used for getting the parameter values in the closed loop condition by employing the controller auto tuning technique, also available in MATLAB. The parameters thus obtained are shown in TABLE I.

$$\frac{\omega(s)}{V(s)} = \frac{44.68s + 3636}{s^2 + 32.43s + 3969} \quad \text{.....(7)}$$

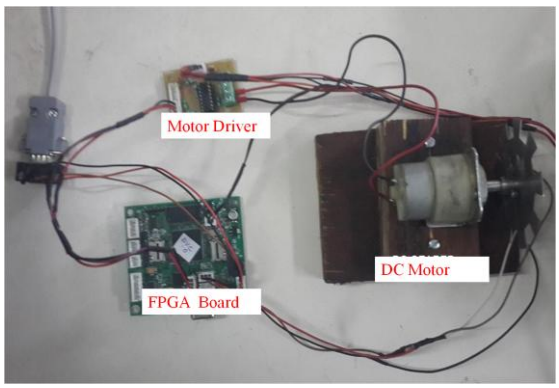

Fig. 3 Experimental Setup

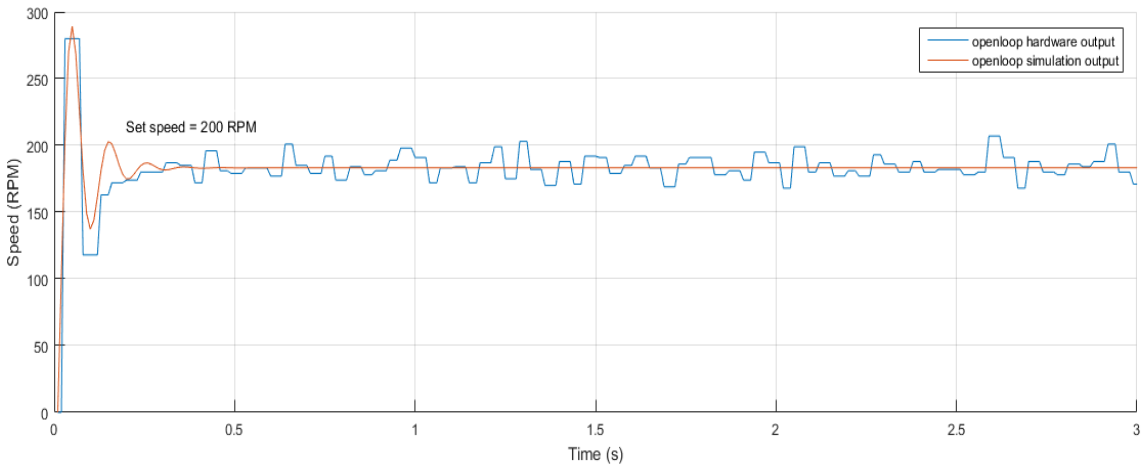

Fig. 4 Open loop response of the hardware and validation of system identification

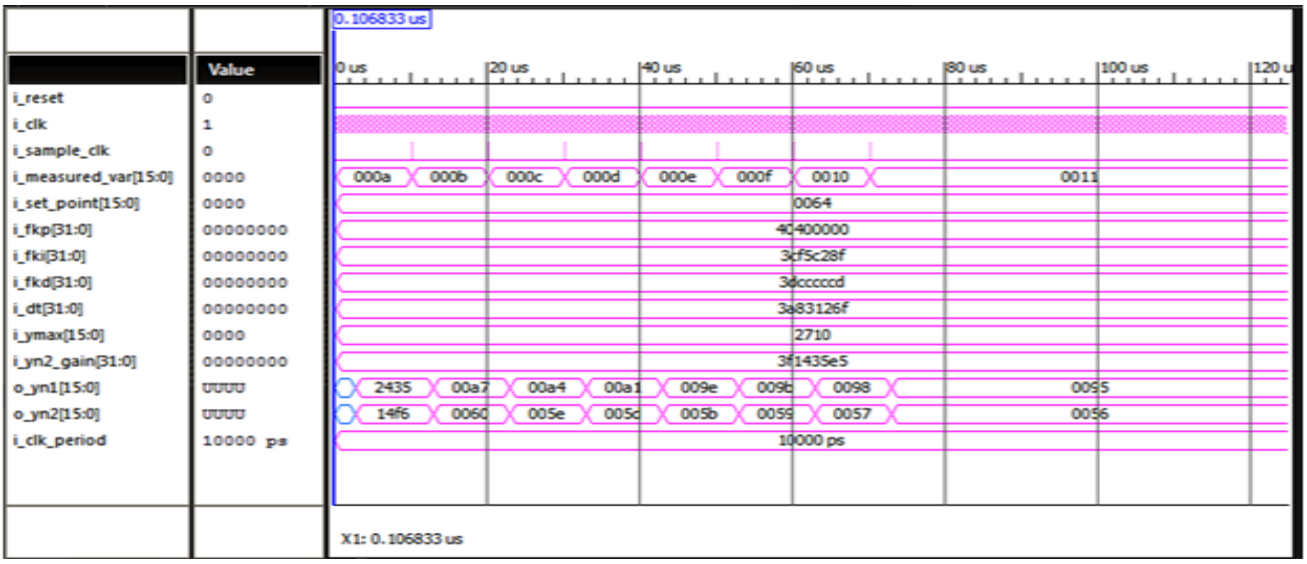

Fig. 5 VHDL Simulation output of proposed controller

TABLE I CONTROLLER PARAMETERS

Tuning Parameters	Values
K	1.8449
T_i	88.6060
T_d	0.00012
N	14806.730

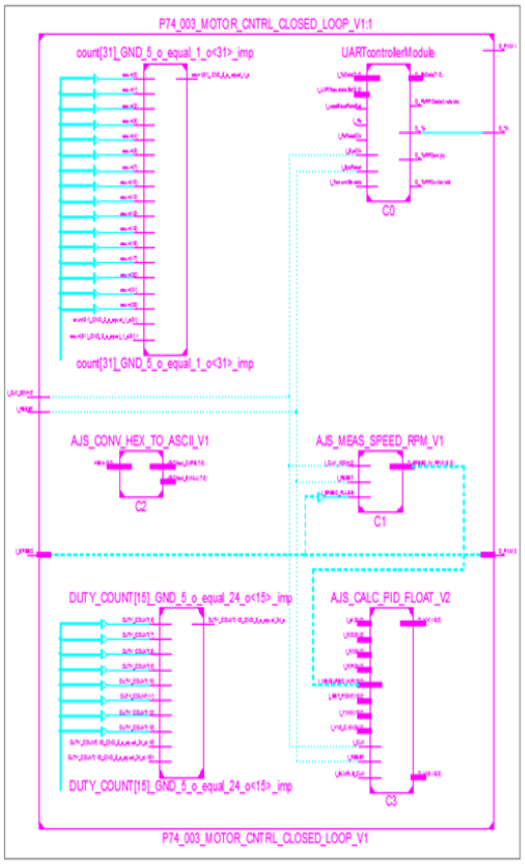

Fig. 6 RTL Schematic of the proposed Controller

V. SIMULATION OF THE PID CONTROLLER

To ascertain the synthesizability of the configuration of the controller, it was simulated in ISim environment by writing its VHDL code and also the code for test vector generation, needed for obtaining the simulated hardware output. VHDL has good abstraction at higher levels than Verilog. [17]. The test vectors, namely, set point variable, measured variable are listed in column 1 of Fig. 5. The output of the FPGA thus obtained is shown in the same figure in terms of hexadecimal value as o_yn1. The generated blocks are shown in Fig. 6 at the register transfer level (RTL). It shows the design abstraction of the FPGA in terms of Register Transfer Level. The control system also includes an UART_controller module for serial communication, a HEX to ASCII module, as shown in the RTL diagram in this figure.

The utilization summary of the device after simulation of the controller for the selected FPGA Spartan 6 LX-9 is as shown in TABLE II.

TABLE II Device Utilization

Logic Blocks	Available	Utilization (%)
Number of Slice Registers	11440	9
Number of Slice LUTs	5720	29
Number of fully used LUT-FF pairs	2039	34
Number of bonded IOBs	102	5
Number of DSP	16	25

VI. RESULTS OBTAINED ON THE COMPLETE HARDWARE UNIT

It is to be noted that the number of pulses used for calculating the speed of the motor is a whole number in 16 bits as measured by optical encoder, and so a separate analog to digital converter is not required in the hardware system. From this encoder output, the speed calculator module computes the speed in RPM using equation (1) which is written as VHDL code in the FPGA. For its further action, the calculated values in hexadecimal are captured and converted into rotor speed in RPM. Then for evaluating the intended functioning of the hardware unit, firstly the simulated closed loop response of the DC motor was studied using MATLAB and then the experimental response was also obtained for a set speed of 150 RPM, both in clockwise and anticlockwise directions. These results are shown in Fig. 7. It is seen that the experiments results are in close agreement with the simulated values. It is also observed, that a minimum settling time of about 5s is required for this chosen system to reach the set speed.

It has been, mentioned earlier in section IV, that there will be a steady state error in the final speed obtained by the DC motor, if it is operated in the open loop mode. It is to be shown that this error can be rectified if a closed loop operation is adopted, even if the set speed is varied frequently, of course keeping in view that the settling time of 5s is required for the present system. The responses of the experimental DC motor, for this kind of servo tracking, obtained for several set speeds both in open loop and closed loop operation are shown in fig. 8 and 9 respectively and compared. Ii is seen that there is an error between final speed and set speed, at every set speed in the open loop system, whereas in the closed loop, this error is negligible. Thus the closed loop controller actuates the DC motor at desired speed which is essentially needed for small scale WMR operations.

VII. CONCLUSION

A closed loop embedded controller has been designed and implemented on FPGA Spartan LX9, for the closed loop speed control of small range DC motors. The hardware utilisation summary of the FPGA reveals that only a small percentage of the available logic blocks inside the FPGA unit are utilised, which gives space for any additional algorithms to be implemented in the hardware for further enhanced working of the system. The necessary general expressions for the design of the controller and detailed procedure involved in the development of this embedded controller are presented. The various performance indices have been evaluated and it is shown that closed loop operation, motor attains the set speed with negligible error compared to that obtained in the open loop operation. UART communication interface and glitch filter are also implemented along with the controller on the same FPGA which makes it an efficient plug and play device. The setpoint tracking obtained successfully for speeds of 50, 100, 150, 175, 125,75 RPM has also been shown which brings out the efficacy of the proposed closed loop controller.

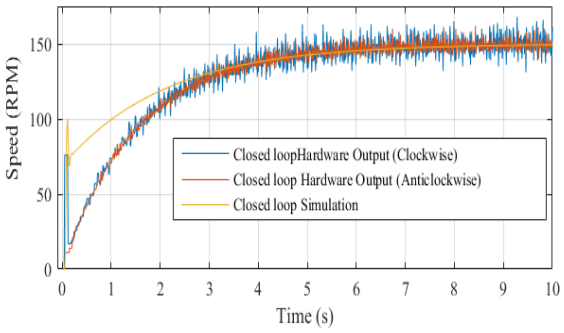

Fig. 7 Simulation and hardware output of the DC motor with controller for one set speed

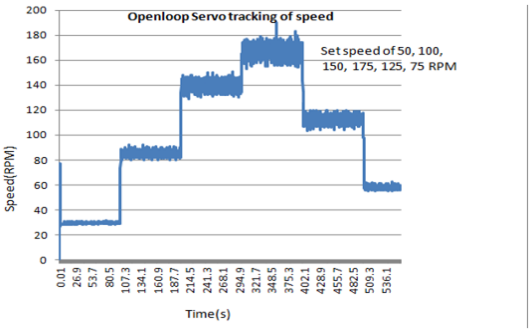

Fig. 8 Hardware output of the DC motor in open loop for various set speeds

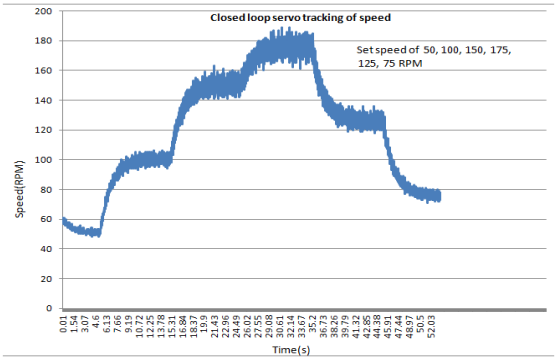

Fig. 9 Hardware output of the DC motor in closed loop for various set speeds

ACKNOWLEDGEMENT

This work is supported in part by the Government of India, UGC minor project grant scheme Ref. No. F. No. 4-4/2015-16 (MRP/UGC SERO).

REFERENCES

[1] Y. Cui, R. M. Voyles, and M. H. Mahoor, "ReFrESH: A self-adaptive architecture for autonomous embedded systems," *IEEE Int. Conf. Autom. Sci. Eng.*, pp. 850–855, 2013.

[2] E. Fitzgerald, C. Kingsley, and S. D. Umans, "Electric Machinery", Tata Mc Graw Hill, *Sixth Edition*.

[3] T. Uzunoviü, E. Žuniü, A. Badnjeviü, I. Miokoviü, and S. Konjicija, "Implementation of digital PID controller," MIPRO, pp. 1357–1361, 2010.

[4] N. Thanh, K. N. Thanh, C. N. The, P. P. Hung, and H. H. Xuan, "Development of Fuzzy Logic Controller for DC Motor Using Personal Computer and Inexpensive Microcontroller," *13th International Conference on Control Automation Robotics & Vision (ICARCV)*, Singapore, pp. 1310-1314, December, 2014.

[5] J. Dong, X. Yang, Q. Liu, Z. Wang and T. Wang, "Design and implementation of CNC controllers using reconfigurable hardware," *IEEE International Conference on Control and Automation*, Christchurch, pp. 1481-1486, 2009.

[6] Y. F. Chan, M. Moallem, and W. Wang, "Design and Implementation of Modular," *IEEE Trans. On Ind. Electronics*, vol. 54, no. 4, pp. 1898–1906, 2007.

[7] A. Astarloa, U. Bidarte, J. Lázaro, J. Andreu, and J. L. Martín, "Configurable-System-on-Programmable-Chip for power electronics control applications," *Proc. - 2008 Int. Conf. Reconfigurable Comput. FPGAs*, pp. 169–174, 2008.

[8] A. Thangavelu, M. V. Varghese, and M. V. Vaidyan, "Novel FPGA based controller design platform for DC-DC buck converter using HDL Co-simulator and Xilinx System Generator," *ISIEA 2012 - IEEE Symp. Ind. Electron. Appl.*, pp. 270–274, 2012.

[9] S. Verma, K. Rajnish, D. Soni, H. Patel, and R. Singh, "Real-Time Controller for Research and Development on ITER Ion Cyclotron Heating and Current Drive Source," *IEEE Transactions on Nuclear Science*, vol. 65, no. 2, pp. 814-820, 2018.

[10] S. Hauck and D. André, "Reconfigurable Computing - The Theory and practice of FPGA-based computing", *Elsevier*, Vol.1, 1st Edition, 2007.

[11] V. Subasri, K. Lavanya, and B. Umamaheswari, "Implementation of Digital PID controller in Field Programmable Gate Array (FPGA)," *2006 India Int. Conf. Power Electron.*, pp. 172–176, 2006.

[12] S. Sánchez-Solano, a J. Cabrera, I. Baturone, F. J. Moreno-Velo, and M. Brox, "FPGA implementation of embedded fuzzy controllers for robotic applications," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1937–1945, 2007.

[13] S. J. S. Jung and S. S. K. S. S. Kim, "Hardware Implementation of a Real-Time Neural Network Controller With a DSP and an FPGA for Nonlinear Systems," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 265–271, 2007.

[14] M. Á. Martínez Prado, J. Rodríguez Reséndiz, D. C. Toledo Pérez, C. M. Torres Hernández, and G. Herrera Ruiz, "Motion Control with FPGA," *IntechOpen*, pp. 57–82, 2017.

[15] R. Cumplido, S. Jones, R. M. Goodall, and S. Bateman, "A High-Performance Processor for Embedded Real-Time Control," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 3, pp. 485-492, 2005.

[16] M. Kocur, S. Kozak, and B. Dvorscak, "Design and Implementation of FPGA - Digital Based PID Controller," *15th International Carpathian Control Conference (ICCC)*, pp. 233–236, 2014.

[17] Z. A. Obaid, N. Sulaiman, M. H. Marhaban, and M. N. Hamidon, "Design and Implementation of FPGA-Based Systems -A Review," *Aust. J. Basic Appl. Sci.*, vol. 3, no. 4, pp. 3575–3596, 2009.

[18] K. Astrom, "PID controllers: Theory, design and tuning," *Instrument Society of America, 2nd Edition*, 1995.