Indonesian Journal of Electrical Engineering and Computer Science
Vol. 9, No. 1, January 2018, pp. 204~211
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v9.i1.pp204-211

Codesign Methodology based FPGA and Embedded Linux for Motor Control

Ibtihel Jaziri*, Lotfi Charaabi, Khaled Jelassi
Université de Tunis El Manar, Ecole Nationale d'Ingénieurs de Tunis, LR11ES15 Laboratoire des Systèmes Electriques, 1002, Tunis, Tunisie

Article Info

Article history:

Received Aug 30, 2017
Revised Nov 17, 2017
Accepted Dec 2, 2017

Keywords:

Closed loop control
Codesign methodology
Dynamic reconfiguration
Embedded Linux
FPGA

ABSTRACT

Recently, the complexity of the embedded electrical systems is increasing due to the growing of industrial necessities. To handle with this complexity, the use of reconfigurable hardware/software codesign methodology using digital platforms becomes necessary. Reconfigurable methodology can be seen as an essential feature in motor control systems as it offers many advantages such as increased flexibility, low energy consumption. In these kind of systems, both hardware and software designs can be reconfigurated depending upon the designer requirements. This paper presents hardware/software codesign methodology with flexible hardware devices and configurable graphical user interface: the hardware architecture is based on field-programmable gate array and the software architecture is based on embedded Linux operating system running on Beaglebone. The user interface is designed as a graphical user interface developed on an emebdded server; this enables the user to access to the system rapidly, parameterize and observe the behavior of the controller.

Corresponding Author:

Ibtihel Jaziri,
Department of Université de Tunis El Manar, Ecole Nationale d'Ingénieurs de Tunis,
LR11ES15 Laboratoire des Systèmes Electriques, 1002, Tunis, Tunisie
Email: ibtihel.jaziri@enit.utm.tn

1. INTRODUCTION

Embedded control systems are gradually becoming an essential aspect of many industrial applications. Nowadays, these systems become more and more sophisticated due to the growing complexity of control algorithms and the rising of industrial requirements [1]. Those requirements are not limited on a high-level of performances. Indeed, flexibility, reliability and low cost are also of prime importance [2]. In field of power electronics and drive applications, these systems are characterized by high performance requirements such as high integration, flexibility and efficiency. The use of efficient reconfigurable codesign methodologies, which exploit the advantages of the current digital technologies, offer an ideal solution to deal with this complexity [3]-[6]. Dynamic reconfiguration can be seen as an integral feature of embedded control systems. It enables system modification at run-time: thus, designers can change the executing applications on these systems [7]. Depending on the application demands, designer has to choose the best solution among these digital technologies. Today, many complex algorithms are implemented using software digital devices such as digital signal processors (DSPs) or microcontrollers due to their software flexibility and low cost, however, these devices still limited for complex structures. The more complex is the control algorithm, the longer is the execution time [8]. In the recent control applications, the full hardware devices such as FPGAs which can be considered as an appropriate solution to implement complex algorithms in industrial applications due to their reconfigurable nature, flexibility, high-density functionality, high performance and low-cost development [9]-[11]. It allows designers to develop a hardware architecture that is fully dedicated to the algorithm to implement. Their high integration rate and their ability to exploit the inherent parallelism of the implemented algorithm make them more advantageous compared to pure software platform such as microcontrollers and digital signal processors (DSP) [12]. More recently, it has also become typical to use mixed hardware/software devices which are the combination between the hardware treatment allowed by FPGA and the software treatment ensured by the single board's microcontrollers such as Arduino, Raspberry pi and Beaglebone. Indeed, the use of such devices is become a new trend. Due to their numerous advantages, such as low energy consumption, powerful computing, capacity and small size, these boards are omnipresent nowadays [13]. They are well adapted for diverse application domains like robotics, control systems, Linux computing server and even more [14].

These last devices have a positive impact on the development of an efficient codesign solution with the use of an operating system. Indeed, it offers many advantages to control the complexity of embedded control systems such as contribution of multitasking, easy task scheduling, real time execution [15]. Additionally, it allows more flexibility, since it is possible to implement the software part as one or multiple user space processes. As consequence, the use of FPGA and single board computer running embedded Linux operating system is highly appropriate to reach an optimal codesign solution between the algorithm complexity, the required performances and the good flexibility.

In this paper, authors propose dynamically reconfigurable HW/SW codesign approach for DC motor control. The codesign process is based on two reference designs: the first one is a full software implementation of a speed control loop using Beaglebone board running an embedded Linux operating system and the second one is a full hardware implementation of the same control loop using FPGA Xilinx Spartan 3E. The main contribution of this work is to present a low-cost web-based motor control with a graphical user interface (GUI) embeddable in web pages running in a low-cost single board computer Beaglebone that deploy the server and controller and reconfigure the FPGA remotely by charging the full hardware implementation via the designed web page. The proposed HW/SW codesign approach was tested on a PI based speed controller of a DC motor.

This paper is organized as follows: Section 2 describes the proposed codesign methodology. The full hardware FPGA-based implementation is detailed in section 3. Section 4 presents the full software implementation. Section 5 describes the reconfigurable codesign methodology. FPGA reconfigurable web interface is descripted in section 6. The experimental results are presented in section 7. Conclusions are given in section 8.

2. RESEARCH METHOD

Figure 1 presents the proposed codesign approach for DC motor control. The principle is to control the plant by using two solutions which can be implemented on hardware and software.

In our reconfigurable codesign system, both the hardware and software parts can be reconfigurated depending upon the designer requirements. The control architecture was split in two parts: hardware and software. The hardware part, ensured by the FPGA technologies, integrates three IP building blocks for PI-based speed control: PI controller, PWM, Speed capture. The software part, allowed by an embedded Linux operating system, includes the same modules developed under embedded Linux. In our codesign system, the hardware portion is specified using a hardware description language such as very high speed integrated circuit hardware description language (VHDL) and the software portion is written in C.

In short, Our strategy consists in using low-cost single board namely Beaglebone based on embedded Linux operating system,firstly, to deploy the PI-based controller application to control DC motor, secondely, to deploy the web server that connects the GUI to the controller and finally to reconfigurate the FPGA through embedded Linux web server. This solution aims to maximize the performances of the controlled system.

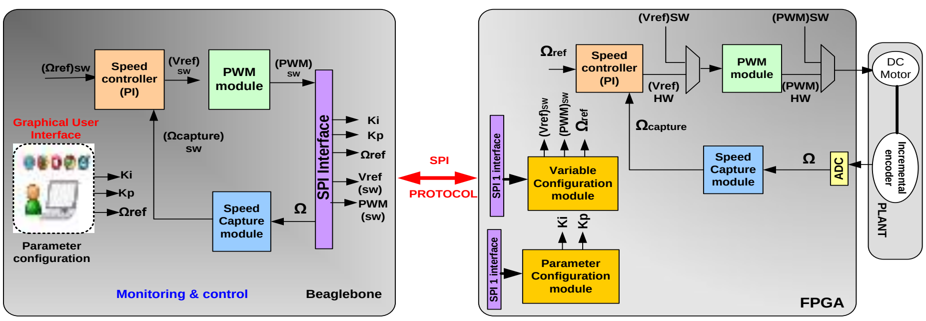


Figure 1. Proposed codesign methodology

3. RESULTS AND ANALYSIS

3.1. FPGA-based hardware implementation

The full hardware implementation is achieved by developing the FPGA architecture dedicated to the speed control algorithm. Indeed, FPGA has already been successfully applied in a wide range of power electronics and drive applications such as control motor [16]-[18] and pulse width modulation PWM [19],[20], Direct torque control of induction motor [21] and field oriented control (FOC) drives [22]-[24].

The developed architecture corresponding to the speed control algorithm for DC motor results from a modular partitioning which divides the corresponding algorithm in four modules namely: 1) the speed capture module, 2) the PI controller module, 3) the PWM module and 4) SPI interface module. The developed speed controller architecture is coded with a hardware description language as VHDL. The designer can develop a fully dedicated FPGA architecture that preserves the algorithm parallelism. Consequently, the execution time is reduced.

3.1.1. Modular partitioning

The algorithm modular partitioning aims to decompose the whole control algorithm into subparts called modules with different level of granularity. These modules are independent and reusable [25]. As presented in Figure 2, our proposed control algorithm can be divided into three levels of granularity. The first level corresponds to the scalar arithmetic operators such as multipliers, adders…The middle level contains the modules used in the control of the DC motor such as anti- windup PI controller, PWM, speed capture. Finally the whole algorithms constitute the third hierarchical level of the library. All of HW modules of this library are written in VHDL.

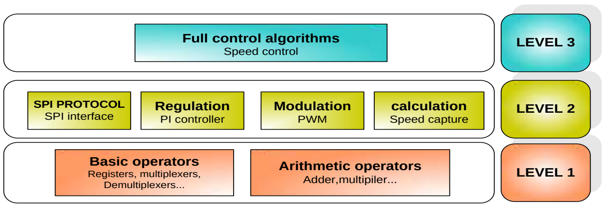
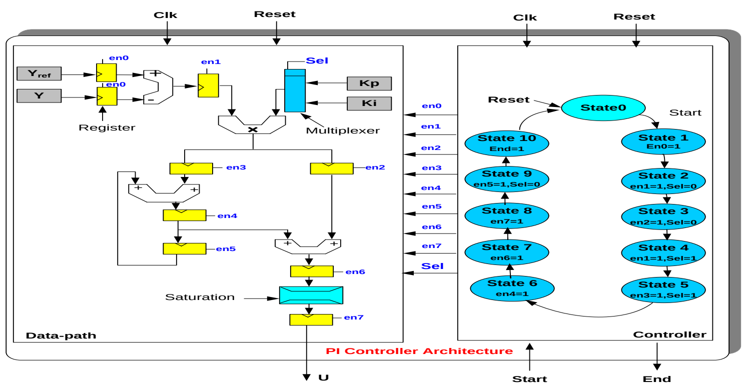


Figure 2. Algorithm modular partitioning

3.1.2. Design of the modular hardware architecture

There are several modules used in the PI-based speed control algorithm for DC motor. For the PI controller module, an appropriate data path and control unit is established with the help of the A3 factorization. The data path is composed of elementary operators which have been presented as the first hierarchical level in the modular partitioning such as adder, multiplier, register. The data transfers between these basic operators are managed by a control unit, which is synchronized with the clock signal. Figure 3 presents the factorized data flow graph (FDFG) of the PI controller based on A3 methodology. The data transfers between the different operators are managed by a control unit which is synchronized with the clock signal (Clk). The control unit of this module is activated via a Start signal. The End pulse signal indicates to the control unit that the data output of the PI controller module is ready to be used.



(a) (b)

Figure 3. PI controller hardware architecture (a) Data path and (b) Control unit

3.1.3. Design Synthesis and Time/area Performance Analysis

The design of the hardware architecture was implemented in a low cost Spartan 3-E XC3S1200E from XLINX. This FPGA contains 400,000 logic gates and includes an internal oscillator that delivers a clock frequency 50 MHz. The designed hardware architecture is shown in Figure 5. It is coded in VHDL hardware description language. It was based on an efficient methodology brings in considerable design advantages such as reusability, optimization of the consumed resources and reduction of the execution time. The control algorithm results from a modular partitioning which includes several modules. As shown in Figure 4 (a), the control algorithm includes a speed capture module, a PI controller module, a PWM module and SPI protocol module. An algorithm controller realizes the data transfer and the management between these different modules. Firstly, the control unit activates the speed capture module that computes the mesured speed Ω_{capt}. The computation time of this module is equal to 0.3 µs. When the computation process of the speed capture module is achieved, the control unit activates the SPI protocol module that computes the PI controller parameters Ki and K_p with a computation time equal to 0.32µs. When the computation process of the SPI module is completed, the PI controller module is activated, it computes the voltage reference Vref needed to generate the PWM signals. The computation time of this module is equal to 0.28µs. Finally, the control unit activates the PWM module that computes the duty cycle U. The whole designed hardware architecture is managed via the global control unit which is simple FSM which the proposed control algorithm. It is activated via an active Start pulse signal. A new computation cycle is activated at the next start active pulse. This leads to the sequential timing diagram presented in Figure 4 (b). In this figure, the whole execution time is the sum of the different modules discussed above.

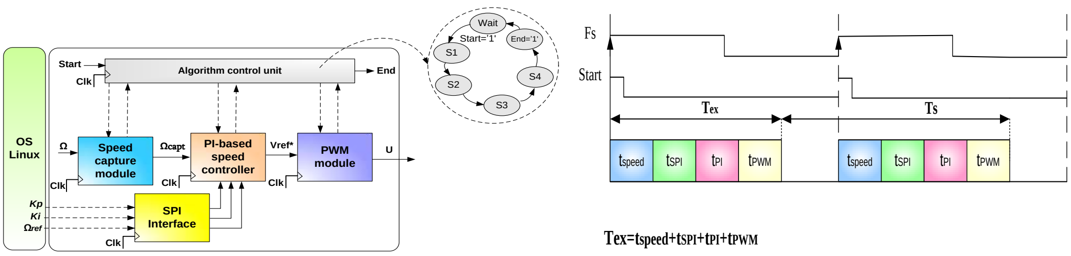


Figure 4. (a) Hardware architecture of the FPGA-based DC motor control (b) Timing diagram of the designed hardware architecture

The computation process is activated every sampling period Ts which is greater than the execution time. The FPGA time/area performances of the considered PI-based speed controller algorithm of DC motor are presented in Table II. The whole execution time Tex is 0.98 µs. The consumed resources of Table 1 are obtained for a 12 bit fixed point format.

Table 1. FPGA Time/Area Performances Of The Hardware Architecture

Modules	Latency	Computation Time
Speed capture	15	Tspeed=0.3µs
SPI interface	16	TSPI=0.32µs
PI controller	14	TPIcontroller=0.28µs
PWM	4	TPWM=0.08µs
Execution time $T_{ex}=t_{speed}+t_{SPI}+t_{PIcontroller}+t_{PWM}$		0.98µs

3.2. Software-based implementation

Nowadays, a wide range of control applications are mostly carried out with software-based controllers such as Beaglebone, Raspberry pi and Arduino. The main reasons of this statement are: low cost of the device, Easy way of coding (C/C++), good flexibility, rapid prototyping and the high level of integration since a Beaglebone integrates many heterogeneous elements in the same device (PWM, analog-to-digital converter (ADC), Serial Protocol Interface (SPI)…).

The Beaglebone board has many interesting Inputs/Outputs capabilities, such as USB client and host, Ethernet, HDMI, GPIO with PWM and even built-in support for I2C and SPI [13]. Beaglebone is chosen because it is a low cost and open hardware platform [14]. The speed control application and the embedded server are implemented in the Beaglebone. Regarding the controller application, it is deployed in the Beaglebone and is in charge of the speed control loop over the plant. To perform both tasks, we implement a C application. On one hand, this application is in charge of closing the control loop over the plant and sends to the server the values of those signals. On the other, it receives the parameterization messages generated by the user' interaction with the Graphical user interface (GUI) sent through the web server to the controller. The communication of the controller application and the sensors used in the plant can take advantage with the use of General purpose Input/output (GPIO) Beaglebone's pins that allows to have a physical communication between the board and the plant. This interface is used as a classical input/output board, where C functions are available to read and write on the corresponding input/output channels.

3.3. Reconfigurable Codesign methodology

To reach a rigorous reconfigurable HW/SW codesign methodology, the use of an embedded web server, is become necessary. His main role of is to ensure communication between the user and the plant.

3.3.1. Embedded web server

The embedded web server makes the link between the plant and the user interface by controlling the plant and allowing bidirectional communication. The use of embedded web server for the control of DC motor will provide advantages like cost effectiveness because the required hardware is cheaper when compared to using a PC server. In our proposed contribution, we have chosen the LAMP server which is the combination of Linux, Apache, MySQL and PHP [30]. The LAMP has two characteristics of cost-free and open source that can reduce the Web control system's cost. Embedded web server use the HTTP standard protocol to transmit web pages from the embedded system to the web browser, and to transmit HTML form data from the browser back to the device. The device requires a network interface such as Ethernet, TCP/IP software.

3.3.2. The designed graphical user interface

The graphical user interface is what the user can see in the web browser during experimental phase. This GUI is implemented with HTML, CSS, JavaScript, and JQuery. HTML and CSS is used in the combination to mark up and style the web page. All this software tools are freeware open-source tools for rapid creation of applications with high-level graphical capabilities and with an increased degree of interactivity. Using this applet, the user is able to: 1) switch between the hardware design, software design or mixed design to work with as shown in Figure 5, 2) Send messages with the values of the PWM duty cycle or PI controller parameters (proportional/integral constants K_p/K_i) that are selected in the GUI to the server. The server will then communicate with the plant and send back the feedback the file with the data collected from the plant. 3) Receive evolution signal messages and represents their values in real-time graphical display. 4) Upload VHDL file available on the computer and download it in the FPGA configuration memory. This feature makes the DC motor control web interface more attractive, realistic, and user-friendly.

Figure 5. DC motor control web interface

3.3.3. FPGA reconfigurable web interface

Using remote reconfiguration for FPGA platform has a number of important advantages seeing that it provides a greater accessibility [26]. In our approach, the basic idea is to use the Beaglebone to reconfigure the FPGA using the FPGA's JTAG port to accomplish this task. JTAG has been selected because it is supported by most of the commercially available FPGA devices. A software implementation of a JTAG controller, provided by Xilinx, runs on the Beaglebone and acts as the FPGA programmer. The JTAG configuration program running on the Beaglebone uses Xilinx specific Boundary scan configuration (.xsvf). The Beaglebone runs software that can take the programming file and send it over some I/O pins that are connected to the target device JTAG interface. It shares 6 ports for communication to make possible the reconfiguration from a user web page developed under the embedded web server implemented on Beaglebone as shown in Figure 6.

Figure 6. FPGA reconfigurable web interface

3.3.4. Experimental set up

The structure of the developed control system is shown in Figure 7. The power stage consists of a DC motor linked to an incremental encoder. The digital control unit consists of an FPGA on one hand and Beaglebone running an embedded Linux operating system on the other hand. The following devices are chosen: FPGA Spartan-3E (xc3s1200E) with a 50-Mhz clocking resources and Beaglebone Rev A6.

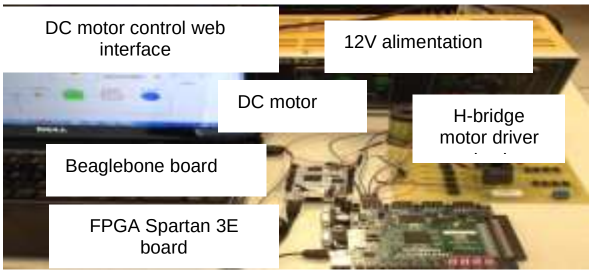


Figure 7. Experimental set up

During the development of the speed-based DC motor control, the benefit of the proposed solution is the reconfiguration of the DC motor speed controller using the embedded web server. Under this latter, a GUI of the DC motor control using software implementation has been developed and presented in Figure 8 where the speed response based on PI controller is displayed. Indeed, this GUI establishes the connection with the server and receives the evolution signal according to the value of PI parameters selected in the closed loop DC motor control web interface.

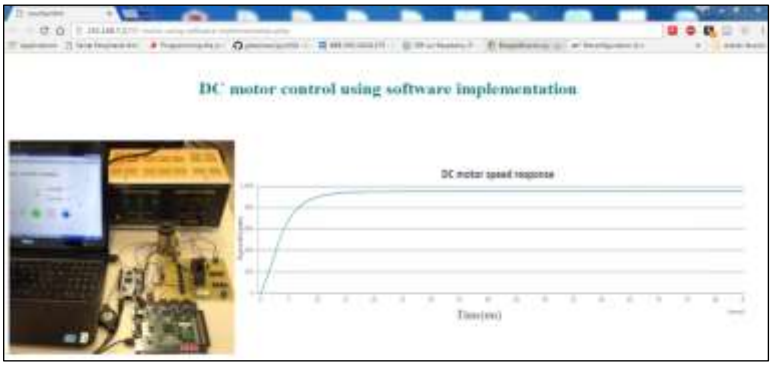


Figure 8. Speed response web interface

4. CONCLUSION

The aim of this paper was to present a dynamically reconfigurable HW/SW codesign methodology for DC motor control based on the use of FPGA and Beaglebone board running embedded Linux operating system. This approach is highly appropriate to reach an optimal compromise between the control performances and the design flexibility. The most salient feature of this approach lies in an embedded web server ensuring the dynamic reconfiguration. This embedded web lets us: develop a GUI to control DC motor by fixing the PI parameters, get a real-time speed response over the selected plant and dynamically reconfigurate FPGA.

The proposed solution reduces the communication delays between the different elements by placing the server and controller in the same low-cost single-board computer. In the future work, authors are going to propose a HW/SW co-design methodology of a sensorless control for DC motor using Luenberger observer based on FPGA and embedded Linux including the speed control loop and current control strategy will be achieved.

REFERENCES

[1] B. Kienhuis, Ed F. Deprettere, P. van der Wolf and K. Vissers . A Methodology to Design Programmable Embedded Systems: The Y-Chart Approach. *Embedded Processor Design Challenges*. April 2002; Vol. 2268: 18-37.

[2] A. Ben Salem, S. Ben Othman, S. Ben Saoud. Field programmable gate array-based system-on-chip for real-time power process control. *International American Journal of Applied Sciences*. 2010: 127–139.

[3] E. Monmasson, L. Idkhajine, I. Bahri, M. W. Naouar and L. Charaabi. *Design methodology and FPGA-based controllers for Power Electronics and drive applications.* 5th IEEE Conference on Industrial Electronics and Applications. 2010 :2328-2338.

[4] I. Bahri, L. Idkhajine, E. Monmasson, M.E.A. Benkhelifa, Optimal hardware/software partitioning of a system on chip FPGA-based sensorless AC drive current controller. *Mathematics and Computers in Simulation*. April 2013; Vol. 9:145-161.

[5] Qi Zhang, Zongwu Xie, Fenglei Ni, Hegao Cai and Hong Liu, *A high performance FPGA-based joint controller with hardware/softwareco-design method*, IEEE International Conference on Mechatronics and Automation. 2012: 1109-1114.

[6] I. Bahri, L. Idkhajine, E. Monmasson and M. El Amine Benkhelifa. Hardware/Software Codesign Guidelines for System on Chip FPGA Based Sensorless AC Drive Applications. *IEEE Transactions on Industrial Informatics*. 2013; Vol.9(4): 2165-2176.

[7] I. Rafiq Quadri, S. Meftali and J.L. Dekeyser. *Designing dynamically reconfigurable SoCs: From UML MARTE modelsto automatic code generation*. Conference on Design and Architectures for Signal and Image Processing (DASIP). 2010: 68-75.

[8] E. Monmasson, L. Idkhajine, M. N. Cirstea, I. Bahri, A. Tisan, M. W. Naouar, and L. Charaabi. *FPGAs in industrial control applications*. IEEE Trans. Ind. Inf. May 2011; vol. 7(2) :224–243.

[9] [9] E. Monmasson, I. Bahri, L. Idkhajine, A. Maalouf and W. M. Naouar. Recent advancements in FPGA-based controllers for AC drives applications. *13th International Conference on Optimization of Electrical and Electronic Equipment (OPTIM)*. 2012: 8-15.

[10] M. W. Naouar, E. Monmasson, A. Ammar Naassani, I. Slama-Belkhodja, and N. Patin. FPGA-Based Current Controllers for AC Machine Drives—A Review. *IEEE transactions in inductrial electronics*. 2007; Vol. 54(4): 1907-1925.

[11] I. Bahri, A. Maalouf, L. Idkhajine and E. Monmasson. FPGA-based implementation of sensorless AC drive controllers for embedded electrical systems. *Symposium on Sensorless Control for Electrical Drives*. 2011: 13-18.
[12] M. W. Naouar, A. A. Naassani, E. Monmasson and I. Slama-Belkhodja. FPGA-Based Predictive Current Controller for Synchronous Machine Speed Drive. *IEEE Transactions on Power Electronics*. 2008; Vol. 23(4): 2115 – 2126.
[13] A. Nayyar and V. Puri. A Review of Beaglebone Smart Board's- A Linux/Android Powered Low Cost Development Platform based on ARM Technology. 9th International Conference on Future Generation Communication and Networking. 2015: 55-63.
[14] S. F. Barrett and J. Kridner. Bad to the Bone: Crafting Electronics Systems with BeagleBone and BeagleBone Black. Morgan and Claypool publishers. April 2013: 1□5.
[15] I. Bahri, M. A. Benkhelifa and E. Monmasson. HW-SW Real-Time Operating system for AC drive applications. *International Symposium on Power Electronics Power Electronics, Electrical Drives, Automation and Motion*. 2012:194-199.
[16] L. Idkhajine, M-W. Naouar, E. Monmasson and A. Prata. Fully FPGA based system on chip solution for current control of AC machine. *European Conference on Power Electronics and Applications*. 2007: 1–10.
[17] I. Bahri, J. Arbi, I. Slama-Belkhodja and E. Monmasson. FPGA-based fault-tolerant current controllers for induction machine. *8th International Symposium on Advanced Electromechanical Motion Systems & Electric Drives Joint Symposium*. 2009:1-6.
[18] Ramesh Balaji S.M., Muniraj C, Mekala N. Wavelet Transform Based Fault Diagnosis of BLDC Motor Drive. TELKOMNIKA Indonesian Journal of Electrical Engineering. June 2015; Vol. 14(3): 434-440.
[19] C.R. Balamurugan, S.P.Natarajan, T.S.Anandhi, R.Bensraj. Hardware Implementation of Cascaded Hybrid MLI with Reduced Switch Count. *Indonesian Journal of Electrical Engineering and Computer Science*. August 2016; Vol. 3(2): 314-322.
[20] R. Rajendran and N. Devarajan. FPGA Implementation of Space Vector PWM Technique for Voltage SourceInverter Fed Induction Motor Drive. *Second International Conference on Computer and Electrical Engineering*. 2009; Vol. 2: 422 – 426.
[21] O. Buchholz and J. Boecker. FPGA-based dynamically reconfigurable control of induction motor drives. *39th Annual Conference of the IEEE Industrial Electronics Society*. 2013: 3543 – 3548.
[22] W.M.Naouar, L. Charaabi, E. Monmasson, I. Slama-Belkhodja. Realization of FPGA reconfigurable IP_core functions for the control of electrical systems. *EPE-PEMC'2004 Conf. Proc.*, Riga, Latvia. September 2004:1-9.
[23] H.R.Khoei, M. Zolfaghari. New Model Reference Adaptive System Speed Observer for Field-Oriented Control Induction Motor Drives Using Neural. *Bulletin of Electrical Engineering and Informatics*. March 2016; Vol. 5(1): 25-36.
[24] L. Charaabi, E. Monmasson, and I. Slama-Belkhodja. Presentation of an efficient design methodology to develop IP-core functions for control systems: Application to the design of an antiwindup PI controller. *IEEE IECON*. 2002: 1942–1947.
[25] Q. Guo, Y. Dai and B. Zhu. Design and implementation of web controller for E-experiment". *IEEE International Conference on Signal Processing Communication and Computing (ICSPCC)*. 2013: 1-4.
[26] Y. Esteves Krasteva, J. Portilla, E. de la Torre and T. Riesgo. Embedded Runtime Reconfigurable Nodes for wireless Sensor Networks Applications. *IEEE Sensors Journal*. 2011; Vol. 11(9): 1800-1810.