Design and Implementation of FPGA-Based Systems -A Review

Article *in* Australian Journal of Basic and Applied Sciences · October 2009

CITATIONS
48

READS
5,767

4 authors:

Nasri Sulaiman
Universiti Putra Malaysia
126 PUBLICATIONS **1,898** CITATIONS
SEE PROFILE

Zeyad Assi Obaid
University of Diyala
30 PUBLICATIONS **205** CITATIONS
SEE PROFILE

Mohammad Hamiruce Marhaban
Universiti Putra Malaysia
190 PUBLICATIONS **1,309** CITATIONS
SEE PROFILE

Mohd Nizar Hamidon
Universiti Putra Malaysia
163 PUBLICATIONS **977** CITATIONS
SEE PROFILE

Australian Journal of Basic and Applied Sciences, 3(4): 3575-3596, 2009
ISSN 1991-8178

Design and Implementation of FPGA-Based Systems - A Review

Nasri Sulaiman, Zeyad Assi Obaid, M. H. Marhaban and M. N. Hamidon

Department of Electrical & Electronic Engineering, Faculty of Engineering, University Putra Malaysia, 43400 UPM Serdang, Selangor Darul Ehsan, Malaysia.

Abstract: This paper reviews the state of the art of field programmable gate array (FPGA) with the focus on FPGA-based systems. The paper starts with an overview of FPGA in the previous literature, after that starts to get an idea about FPGA programming. FPGA-based neural networks also provided in this paper in order to highlight the best advantage by using FPGA with this type of intelligent systems, and a survey of FPGA-based control systems design with different applications. In this paper, we focus on the main differences between software-based systems with respect to FPGA-based systems, and the main features for FPGA technology and its real-time applications. FPGA-based robotics systems design also provided in this review, finally, the most popular simulation results with FPGA design and implementations are highlighted.

Key words: FPGA-based, Control systems, neural networks, robotics systems design, Programming with FPGA, FPGA Design and implementations.

INTRODUCTION

Most of the physical systems applications require a real-time operation to interface high speed constraints. The simple and usual way to implement these systems is to realize it as a software program on general purpose computers, these ways can not be considered as a suitable design solution. Higher density programmable logic device such as FPGA can be used to integrate large amounts of logic in a single IC. FPGA becomes one of the most successful of technologies for developing the systems which require a real time operation. For these systems (Boaz Hirschl and Leonid P. Yaroslavsky), (Dr. Subbarao, 2004), (Eric Monmasson and Marcian N. Cirstea, 2007), (Grout, I.A. and K. Keane), (Richard Wain, Ian Bush, 2006) and (Franjo Plavec,) FPGAs are more sufficient than the simple way because they can cover a much wider range of operating conditions. FPGA are two dimensional arrays of logic blocks and flip-flops with an electrically programmable interconnection between logic blocks. The interconnections consist of electrically programmable switches which is why FPGA differs from Custom ICs, as Custom IC is programmed using integrated circuit fabrication technology to form metal interconnections between logic blocks (Brosch, O., J. Hesser,), (Hans-Peter Röser and Felix Huber,), (Franjo Plavec,). In an FPGA logic blocks are implemented using multiple level low fan in gates, which gives it a more compact design compared to an implementation with two-level AND-OR logic. FPGA provides its user a way to configure (Saar Drimer, 2008): The intersection between the logic blocks and the function of each logic block. Logic block of an FPGA can be configured in such a way that it can provide functionality as simple as that of transistor or as complex as that of a microprocessor. It can used to implement different combinations of combinational and sequential logic functions. Logic blocks of an FPGA can be implemented by any of the following:

1. Transistor pairs.
2. Combinational gates like basic NAND gates or XOR gates.
3. N-input Lookup tables.
4. Multiplexers.
5. Wide fan in And-OR structure.

Routing in FPGAs consists of wire segments of varying lengths which can be interconnected via electrically programmable switches. Density of logic block used in an FPGA depends on length and number of wire segments used for routing. Number of segments used for interconnection typically is a trade off

Corresponding Author: Zeyad Assi Obaid, Department of Electrical & Electronic Engineering, Faculty of Engineering, University Putra Malaysia, 43400 UPM Serdang, Selangor Darul Ehsan, Malaysia.
E-mail: eng.alhamdany@yahoo.com

between density of logic blocks used and amount of area used up for routing. The ability to reconfigure functionality to be implemented on a chip gives a unique advantage to designer who designs his system on an FPGA It reduces the time to market and significantly reduces the cost of production. By the early 1980's large scale integrated circuits (LSI) formed the back bone of most of the logic circuits in major systems Microprocessors, bus/IO controllers, system timers etc were implemented using integrated circuit fabrication technology. Random "glue logic" or interconnects were still required to help connect the large integrated circuits in order to: Generate global control signals (for resets etc.) and Data signals from one subsystem to another sub system (FPGA tutorial, 2008), (Boaz Hirschl and Leonid P. Yaroslavsky), (Dr. Subbarao, 2004), (Daryl Popig, Debra Ryle, 2006), (Jason Villarreal, 2007) and (Oskar Mencer, Marco Platzner, 2001). Systems typically consisted of few large scale integrated components and large number of SSI (small scale integrated circuit) and MSI (medium scale integrated circuit) components. Initial attempt to solve this problem led to development of Custom ICs which were to replace the large amount of interconnect. This reduced system complexity and manufacturing cost, and improved performance. However, custom ICs have their own disadvantages. They are relatively very expensive to develop, and delay introduced for product to market (time to market) because of increased design time. There are two kinds of costs involved in development of Custom ICs (A tradeoff usually exists between the two costs): Cost of development and design and cost of manufacture (FPGA tutorial, 2008). Figure (1) shows the internal architecture of the simplified version of FPGA.

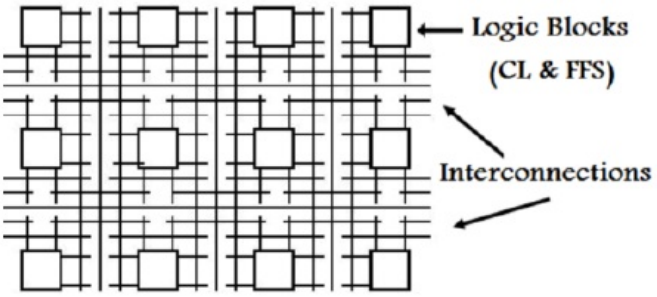

Fig. 1: Simplified version of FPGA internal architecture (FPGA tutorial, 2008).

Therefore the custom IC approach was only viable for products with very high volume, and which were not time to market sensitive was. FPGAs were introduced as an alternative to custom ICs for implementing entire system on one chip and to provide flexibility of reporogramability to the user. Introduction of FPGAs resulted in improvement of density relative to discrete SSI/MSI components (within around 10x of custom ICs). Another advantage of FPGAs over Custom ICs is that with the help of computer aided design (CAD) tools circuits could be implemented in a short amount of time (no physical layout process, no mask making, no IC manufacturing) (Viejo, J., J. Juan, M.J. Bellido, 2008), (Yasuo Sakaide, 1999), (FPGA tutorial, 2008). Figure (2) shows a comparative analysis of FPGA.

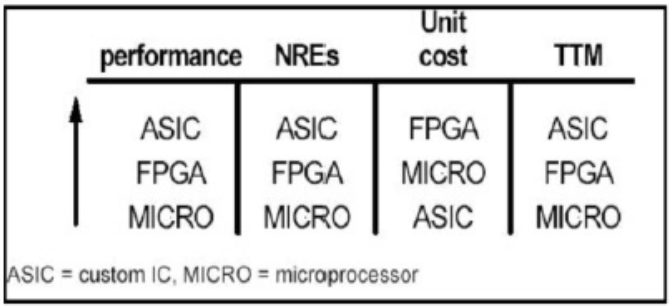

Fig. 2: FPGA comparative analysis (FPGA tutorial, 2008).

II. Programming in FPGA:

The FPGA chip can be programmed using a language called hardware description language (HDL), and this contains two types of the languages, very high description language (VHDL) and verilog language. VHDL is a hardware description language for describing digital designs. It originated from a government programmed in the development of Very High Speed Integrated Circuits. VHDL is like a general programming language

with extensions to model both concurrent and sequential flows of execution and the concept of delayed assignment of values. The code has a very unique structure, which is related to the fact that it is still part of a circuit (Simon Bevan,), (Grout, I.A. and K. Keane), (Robert Trausmuth, 2006). Recent advances in FPGAs have made hardware-accelerated computing a reality for many application domains, including image processing, digital signal processing, data security and communications. Until recently, such platforms required detailed hardware design expertise on the part of the application developer. More recently, software-to-hardware tools have emerged that allow application developers to describe and generate complete software/hardware systems using higher-level languages. This class presents specific techniques for creating FPGA-accelerated applications using the C language (David Pellerin, 2007). Although FPGA technology has been around in some form for many years it is only in the last two to three years that the technology has begun to make any inroads into the HPC market. In the past the vast majority of FPGA users would have been hardware designers with a significant amount of knowledge and experience in circuit design using traditional Hardware Description Languages (HDL) like VHDL or Verilog. These languages and many of the concepts that underpin their use are unfamiliar to the vast majority of software programmers. In order to open up the FPGA market to software programmers, tools vendors are providing an increasing number of somewhat C-like FPGA programming languages and supporting tools. These pseudo-C languages all provide a more familiar development flow that, in many cases, may provide a significant level of abstraction away from the underlying hardware. (Richard Wain, Ian Bush, 2006). Hardware description languages (HDL) were developed to ease the implementation of large digital designs by representing logic as Boolean equations as well as through the use of higher-level semantic constructs found in mainstream computer programming languages. Aside from several proprietary HDLs, the major industry standard languages for logic design are Verilog and VHDL. The two languages have roughly equal market presence. The functionality of a digital circuit can be represented at different levels of abstraction and different HDLs support these levels of abstraction to a greater or lesser extent. The lowest level of abstraction for a digital HDL would be the switch level, which refers to the ability to describe the circuit as a netlist of transistor switches. A slightly higher level of abstraction would be the gate level, which refers to the ability to describe the circuit as a netlist of primitive logic gates and functions. Both switch-level and gate-level netlists may be classed as structural representations. It should be noted, however, that “structural” can have different connotations because it may also be used to refer to a hierarchical block-level netlist in which each block may have its contents specified using any of the levels of abstraction. The next level of HDL sophistication is the ability to support functional representations, which covers a range of constructs.

At the lower end is the capability to describe a function using Boolean equations. The functional level of abstraction also encompasses Register Transfer Level (RTL) representations. The term RTL covers a multitude of manifestations, but the easiest way to wrap one’s brain around the underlying concept is to consider a design formed from a collection of registers linked by combinational logic. The next level of abstraction used by traditional HDLs is known as behavioral, which refers to the ability to describe the behavior of a circuit using abstract constructs like loops and processes. The highest level is a system level of abstraction that features constructs intended for system-level design applications (Maxfield, C., 2004). Descriptions of the development of a prototype software toolbox that can analyze and process a *Simulink* block diagram model in order to produce a *VHDL* representation of the model is presented in (Grout, I.A. and K. Keane). The derived VHDL model will consist of a combination of behavioral, RTL and structural definitions mapped directly from the *Simulink* model. This approach may enable a user to develop and simulate a digital control algorithm using *Matlab* and once complete, convert this to VHDL code. This would then be synthesized into digital logic hardware for implementation on devices such as FPGAs and ASICs (Application Specific Integrated Circuits) (Grout, I.A. and K. Keane), (Jingzhao Ou and Viktor K. Prasanna), (Rebaudengo, M., M. Sonza Reorda,). Different FPGA manufacturers have developed different basic cells, seeking to provide the most useful functionality in the cells for generation of overall FPGA functions. Cells range from fine-grained cells consisting of basic gates through medium-grained cells providing more complex programmable functions to large-grained cells. Different FPGA manufacturers also provide different approaches to the programming step. FPGA programming technologies include one-time programming or multiple-time programming capabilities with the programming either nonvolatile (i.e., programming is retained when power is turned off) or volatile (i.e., programming is lost when power is turned off) (Dorf, R.C., 2000), (Scott Hauck, 1998), (Jason Villarreal, 2007), (Sunghyun Lee, Kiwook Yun, 2001), (Miroslav Vadkerti, 2006).

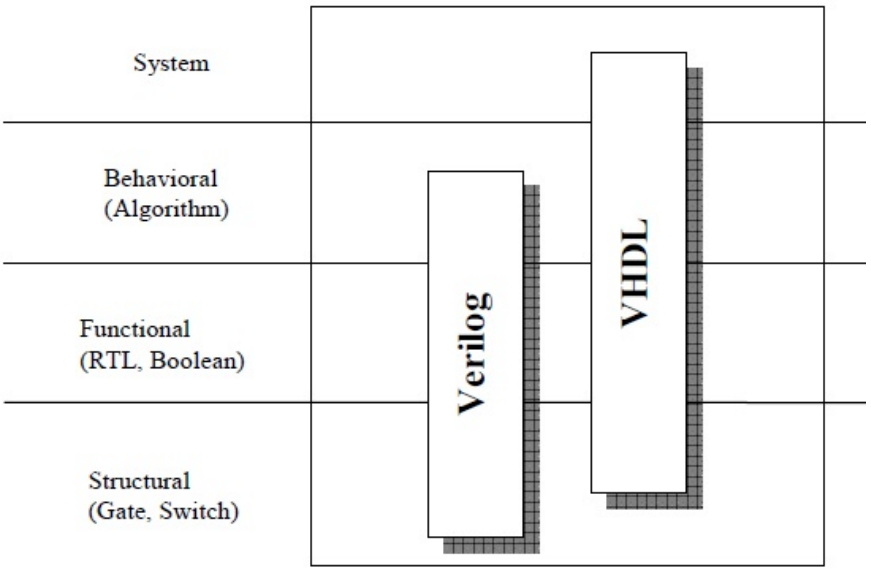

Fig. 3: Levels of abstraction.

III. FPGA-based Neural Networks:

For the present works and researches, FPGA was preferred with the neural networks design and implementations, because it offers high speed constrains and low coast as well as big memory and short time to market, (Dominic Job, Venky Shankararaman 1999). The authors in (Joy, S.P. Vasantha Rani P. Kanagasabapathy) describe design and implementation of a fast and flexible artificial neural network in VHDL and implemented in a Spartan 2E FPGA device family (xc2s300e-6pq208) for testing. The architecture of the neural network is modeled and simulated in VHDL. This neural network is controlled by a single finite state machine (F SM) instead of using separate FSMs for controlling each layer of the neural network thereby reducing the number of gates used. Each neuron has a separate ROM for the storage of weights. Form this design results, the number of gates required for implementing one FSM is 'k' times (approximately) less than that required when the each layer of artificial neural network is controlled by separate FSM. The FPGA can process a three 8-bit input data set every 30 ns, figure (4) shows the synthesized circuit of neural network controlled by FSM which proposed in (Joy, S.P. Vasantha Rani P. Kanagasabapathy).

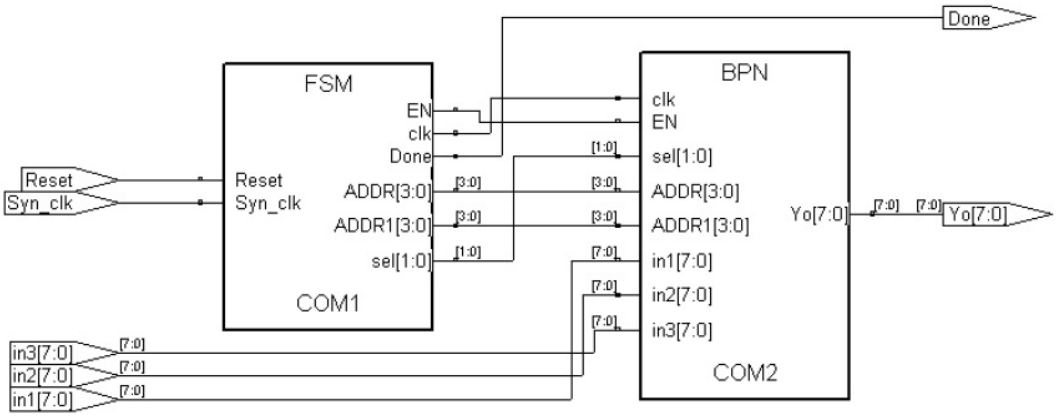

Fig. 4: Synthesized circuit of neural network controlled by FSM.

Savran and Serkan in (Aydo\an Savran and Serkan Ünsal) are presented the hardware implementation of a neural network using FPGA. They designed a Digital system architecture to realize a feed forward multilayer neural network. This design architecture is described using VHDL and implemented in an FPGA chip (Aydo\an Savran and Serkan Ünsal). And for the HW/SW codesign approach for the implementation of multilayer perceptions resulting in an embedded system that can be used in wide variety of applications is presented in

(Alper Ucar and Ali Ziya Alkar). The motivation for the HW/SW co design methodology includes declining time-to-market and power constraints, increasing gap between silicon capacity and computational intensity A finite state machine (FSM) is implemented on FPGA for the synchronization of the feed-forward propagation. The architecture proposed, avoiding on-chip back propagation learning, allows high throughput with low area cost (Alper Ucar and Ali Ziya Alkar). For the neural based instrument prototype in real time application, conventional specific VLSI neural chip design suffers from the limitation in time and cost. With low precision artificial neural network design, FPGAs have higher speed and smaller size for real time application than that of the VLSI design. Novel fully parallel hardware implementations of neural network for EXOR benchmark problem using Xilinx FPGA are presented in (Ali, M.,). The validity of this approach is demonstrated by application to EXOR problem. The design is tested on an FPGA demo board (Ali, M.,). FPGAs are chosen for implement ANNs with the following reason:

- They can implement a wide range of logic gates starting with tens of thousands up to few millions gates.
- They can be reconfigured to change logic function while resident in the system.
- FPGAs have short design cycle that leads to fairly inexpensive logic design.
- FPGAs have parallelism in their nature. Thus, they have parallel computing environment and allows logic cycle design to work parallel.
- They have powerful design, programming and syntheses tools.

So in fully parallel ANN's must be used low number precision (for example 16 bits), the low number precision (16 bit floating point number precision) is suitable for fully parallel network implementation for various applications (Ali, M.,). The authors in (Kai Wang , Van Yuan ,) presented an implementation of Multi-Valued "And/Or"-Neural Network To construct of a multi-layer network in a FPGA and discuss simplified network constructions, they examined the learning algorithm for AND/OR network n-dimensional exclusive-OR problems, and two outputs of 1 bit carry-save-adder, and got correct results. Since the algorithm is not iterative, the calculation is extremely rapid. We calculated the 5-dimensional exclusive-OR problem under 0.05 second on Celeron 300MHz, Windows 98, and Java(version 2)-interpreter (Kai Wang , Van Yuan ,). And also for Kohut and Steinbach in (Roman Kohut and Bernd Steinbach, 2003) are presented a new type of neuron, called Boolean neuron that may be mapped directly to configurable logic blocks (CLBs) of FPGAs. The structure and logic of Boolean neuron allow a direct representation of the Boolean neural network architecture to FPGAs. This approach solves digital design problems especially with respect of the performance and gate count. The additional advantages of Boolean neural networks consist in the reduction of memory space and computation time in comparison to usual neural networks. In the example, and describe the expansion of Boolean output neuron in order to the cascade Boolean neurons with a restricted number of inputs and also the mapping to lookup tables (LUTs) (Roman Kohut and Bernd Steinbach, 2003). Remember, the basic elements of FPGAs are CLBs that contains two LUTs. Each LUT has 4 inputs and 1 output and can realize any Boolean function depending on the 4 Boolean variables.

IV. FPGA with Control Systems Design:

Most of the control systems in the industrial applications required areal-time operation to interface high speed constrains, FPGA is presented as one of the best solutions for this applications, other way like semi-customs and full custom ASIC is presented also to deal with this type of applications, but FPGA offer more flexible design solutions to get of FPGA features (low coast, high speed, short time to market) (Hwu, K.I., Y.T. Tau, 2005), (Roque AlfredoOsornio-Riosa, 2009), (Lieu My Chuong, 2008), (Naylor, G.A., 2003). Correia *et al*. (2008) are presented a description for the implementation of a platform based on reconfigurable architecture and concepts of virtual instrumentation applied to the study of the hands-free driving problem. The novelty of this approach is the use of both reconfigurable systems (for developing the car's controller) and virtual instrumentation issues for developing a high-level abstraction testing and simulation environment. Figure 2 shows the architecture of the control system, which was designed and synthesized using the EDK. The communication of the processor with peripheral devices is achieved by the OPB bus (On-chip Peripheral Bus). There are several hardware peripherals related to the FPGA-based board resources such as display, keyboard, RS232, push-buttons, dip-switches and LEDs. The processor controls the operation flow of the system by running different special designed software functions, which were written in C language and stored in the bRAM-block (see Figure 5) (Anderson P. Correia, 2008).

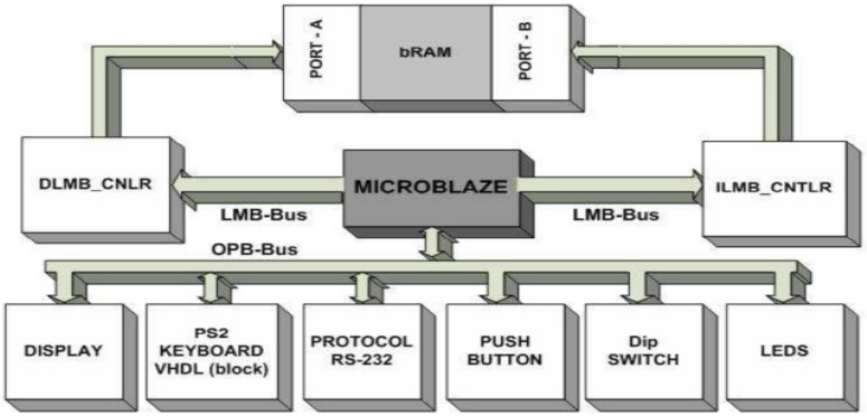

Fig. 5: The Hardware System (Anderson P. Correia, 2008).

Other FPGA- based technique to forward converter is presented in (Hwu, K.I., Y.T. Tau, 2005) to design the PID controller as well as to process the signals from peripherals, to obtain good performance over the entire operating range. They used on-line tuning of controller parameters to reduce the effect of input voltage variations on the transient load response. Besides, the improvement in the transient load response is considered further; especially for the maximum voltage input (Hwu, K.I., Y.T. Tau, 2005). An approach of microstepping control for the step motors based on FPGA is presented in (Xiaodong Zhang, 2005) for improving the control system of some bio-chemistry analysis instrument, and then the relative control system is schematically designed, mainly includes the internal logic design of the FPGA, the Communication interface design of the Microcomputer, the design of the power driving circuit, and the design of the interface circuit between the electrical sources. At last, it is verified by some experimental investigations that the control theory and approach presented in (Xiaodong Zhang, 2005) are corrective, and the designed control system is very good to get a high precision of position control and high control repeatability. So the technology questions from the producing reality are resolved and the product quality is promoted (Xiaodong Zhang, 2005). The authors in (Uffe Jakobsen and Torben Matzen, 2008) described a different approach to hardware and software co-design, namely designing a soft-core processor with an instruction set to fit the purpose of control of drives. Furthermore the soft-core processor is designed with a system for plug in of external logic, motor control for small series sometimes requires specialized control logic, requiring rewiring if new logic needs to be added. Doing so shortens development time, since functionality is simply added to or removed from the soft-core. The designer can then choose between resource usage on the FPGA and execution speed in more degrees. The approach is tested for two different motor types, synchronous and hybrid switched reluctance motors, using a Spartan 3E FPGA. The impact of having ADC-communication in VHDL versus in assembler is also presented (Uffe Jakobsen and Torben Matzen, 2008). The author in (Sornam V. Viswanathan, 2005) discussed the hardware implementation of the digital controller by using FPGA; his work presents Embedded Control Using FPGA.

In automation systems, Proportional-Integral-Derivative (PID) controllers are widely used. Abdelati in (Mohamed Abdelati.) presented the implementation of PID on FPGA board. Several modules necessary for building PID controllers on FPGAs which improve speed, accuracy, power, compactness, and cost effectiveness are outlined. Two PID controllers for speed and position utilizing these modules are implemented and used as experimental platforms to illustrate and test the designed modules (Mohamed Abdelati.). The application of FPGA in high performance DTC induction motor drive is presented in (Jacek Lis, Czeslaw T. Kowalski, 2008), the high performance sensor less AC drives requires a fast digital realization of many mathematical operations concerning control and estimators' algorithms, which are time consuming is presented in (Jacek Lis, Czeslaw T. Kowalski, 2008). Due to the fact that developing an ASIC chip is expensive and laborious, the FPGA based solution should rather be used on the design stage of the algorithm. Few issues concerning the implementation of IM drive control structures in FPGA are discussed and the use of CORDIC algorithm for some mathematical operations in the DTC method is described. Experimental test results of this drive control structure realized in FPGA are demonstrated (Jacek Lis, Czeslaw T. Kowalski, 2008). In this case the control algorithm has to be decomposed into separated parallel tasks (Jacek Lis, Czeslaw T. Kowalski, 2008). And also for power converter control, Pulse width modulation (PWM) has been widely used. Most high power level converters operate at switching frequencies up to 500 kHz, while operating frequencies in excess of 1 MHz at high power levels can be achieved using the planar transformer technology. The contribution of the work

presented in (Eftichios Koutroulis, 2006) is the development of high-frequency PWM generator architecture for power converter control using FPGA and CPLD ICs. The resulting PWM frequency depends on the target FPGA or CPLD device speed grade and the duty cycle resolution requirements. The post-layout timing simulation results are presented, showing that PWM frequencies up to 3.985 MHz can be produced with a duty cycle resolution of 1.56%. Additionally, experimental results are also presented for low cost functional verification of the proposed architecture. Figure (6) shows the Block diagram of the proposed PWM generator in (Eftichios Koutroulis, 2006).

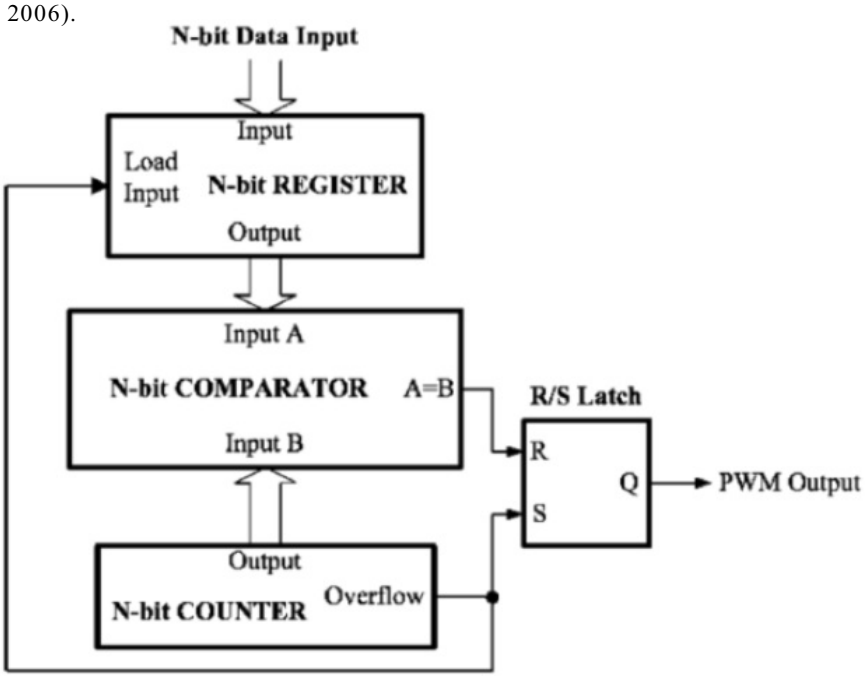

Fig. 6: Block diagram of the proposed PWM generator in (Eftichios Koutroulis, 2006).

V. FPGA-based Robotics Systems Design:

FPGA become in widely used with Robotics applications, since robotics application in many cases required areal-time operation, FPGA offer this type of application more flexibilities with the use of FPGA technologies advantages, high speed, low coast and short time to market. Dynamic Reconfiguration has always constituted a challenge for embedded systems designers, but nowadays, technological developments make possible to do it on Xilinx FPGAs, but setting up a dynamically reconfigurable system remains a painful and complicated task. A framework for performing it in an easy way was proposed in (Andres upegui,), (Benoît R. Veillette and Gordon W. Roberts) for a specific application: Modular Robotics. An architecture containing a Microblaze processor and a reconfigurable module is also proposed during the research in (Andres upegui,). The module is defined in VHDL and synthesized by the user; then to provide the scripts for easily generating the corresponding configuration bit streams for a dynamic partial reconfigurable controller for our Modular Robot. The proposed framework is easily extendable to other applications, the figure below, shows the proposed YaMoR robot in (Andres upegui,).

For implementation of the high-speed CNC Position Controller (PC), the authors in (Yaodong, Tao, Hu lin, 2008) presented an efficient design scheme using FPGA technology. The algorithm is implemented using a Distributed Arithmetic (DA)-based scheme where a Look-Up-Table (LUT) mechanism inside the FPGA is utilized. Two novel DA-based CNC Position Controllers have been proposed for FPGA implementation. The implementation results in (Yaodong, Tao, Hu lin, 2008) show that the two DA-based PCs use 0.8% and 1.5% logic resource of FPGA device respectively comparing the multiplier-based design uses 51.1% logic resource of FPGA device. These two DA-based designs, using a 32 MHz clock as an input clock, can ensure the servo loop update frequency reaches 1 MHz to satisfy the high-speed CNC requirement (Yaodong, Tao, Hu lin, 2008). The description of the technical achievement in developing FPGA-based control system which utilizes the hardware/software re-configurable feature of the advanced FPGA device to achieve the goal is presented in (Narashiman Chakravarthy, 2006). The functional correctness of the closed-loop control system is verified by the experimental tests. The performance analysis shows that the hardware module occupies less FPGA space and the power dissipation is very much comparable to other design alternatives of similar caliber. The

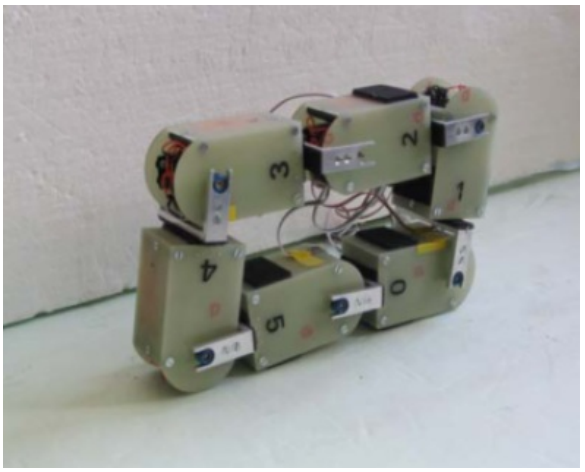

Fig. 7: YaMoR robot: A single unit and a rolling track configuration (Andres upegui,).

preliminary results demonstrate that the PWM-Encoder module, in the form of user Intellectual Property (IP), can be duplicated and re-configured to control as many motors as needed (Narashiman Chakravarthy, 2006), (Andrzej Krasniewski). For passive bipedal robots with complex mechanical structures and multiple degrees of freedoms in body and limbs, Mai et al. in (Jingeng Mai, Qining Wang,) are presented FPGA based high-performance gait control system The system utilizes the hardware and software reconfigurable feature of the advanced FPGA device to achieve intelligent bipedal gait control. And implement central processing unit, sensor data processing module, gait control module, pulse width modulation module and data transmission module inside the FPGA chip. All the modules can be duplicated and reconfigured to control as many motors as needed. In addition, to design a gait control strategy with the method of phase transfer map. To estimate the control system, and also create bipedal robots with passive dynamic gaits. Satisfactory experimental results demonstrate that the FPGA-based gait control system is a flexible, high-performance solution for the locomotion control of passive bipedal robots, figure (8) shows the control system hardware which proposed in (Jingeng Mai, Qining Wang,).

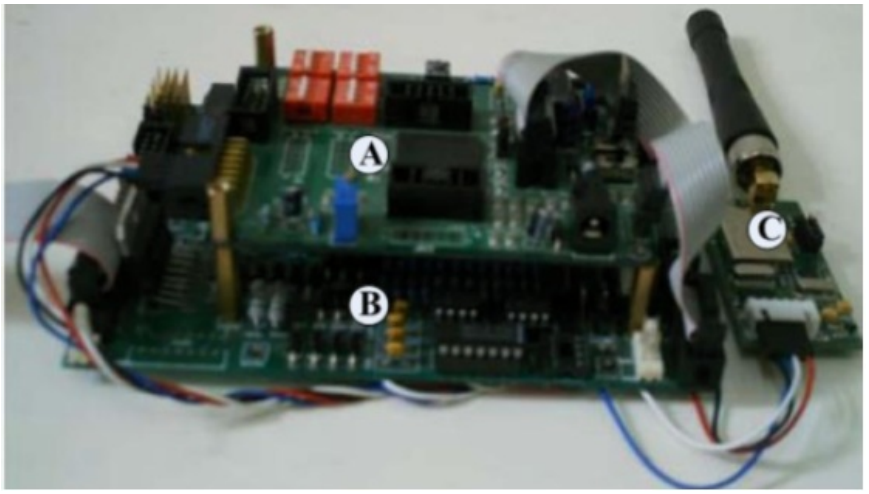

Fig. 8: The picture of the control system hardware (a) is the FPGA-based control board, while (b) is the motor drive board. RF device as shown in (c) is used to transmit data between the control board and remote control computer (Jingeng Mai, Qining Wang,).

VI. FPGA-based Design Results and Simulation:

FPGA has usefully applied to many applications especially with the physical systems, telecommunications, microelectronics, robotics and many of control applications. One of the most exciting innovations in computer architecture in many years is the introduction by several major vendors of FPGA-based processing nodes. FPGAs are fundamentally different from von Neuman (vN) processors: applications are configured in circuitry,

rather than programmed into software. The critical problem to be solved with respect to High Performance Computing using FPGAs (HPC/FPGA) is determining an appropriate method for configuring the FPGAs; this problem involves the often inherent conflict between performance and development cost (Andrzej Krasniewski), (Tom Kean, 2002), (Tom Van Court and Martin C. Herbordt), (Steve Trimberger, 2007), (Wutthikorn Threevithayanon, 2005), (André DeHon and Raphael Rubin, 2004). Figure (15) shows the synthesized results of one neuron, which contains "weightROM" to store the weights of that neuron, "mul" block for multiplying the given input with weight, "Accumulator" to accumulate the input signal from the previous layer and "activation" to threshold the neuron output. Figure (9) gives the simulation output of 3-3-1 neural network with FSM. It is noted that the three 8-bit input data set are processed for every 30 ns with the clock speed of 33.33 MHz (Joy, S.P. Vasantha Rani P. Kanagasabapathy).

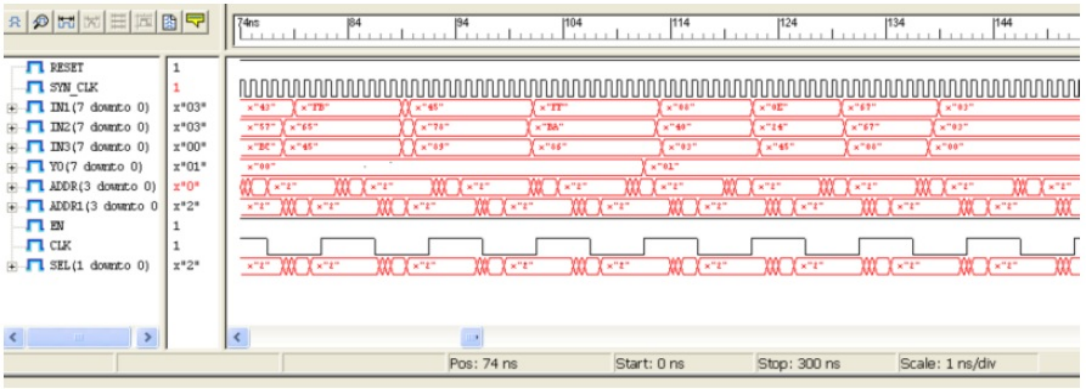

Fig. 9: Neural Network output controlled by FSM (Joy, S.P. Vasantha Rani P. Kanagasabapathy).

In Figure (10-a), is shown the original sound wave (red), the coded sound wave (blue) and the bit stream wave (green) for the coded sound *"You have change"*. In Figure (10-b), a zoom is made to an area of sound in order to see the minimal differences between the original and the coded wave presented in (Gonçalo Martins, 2008).

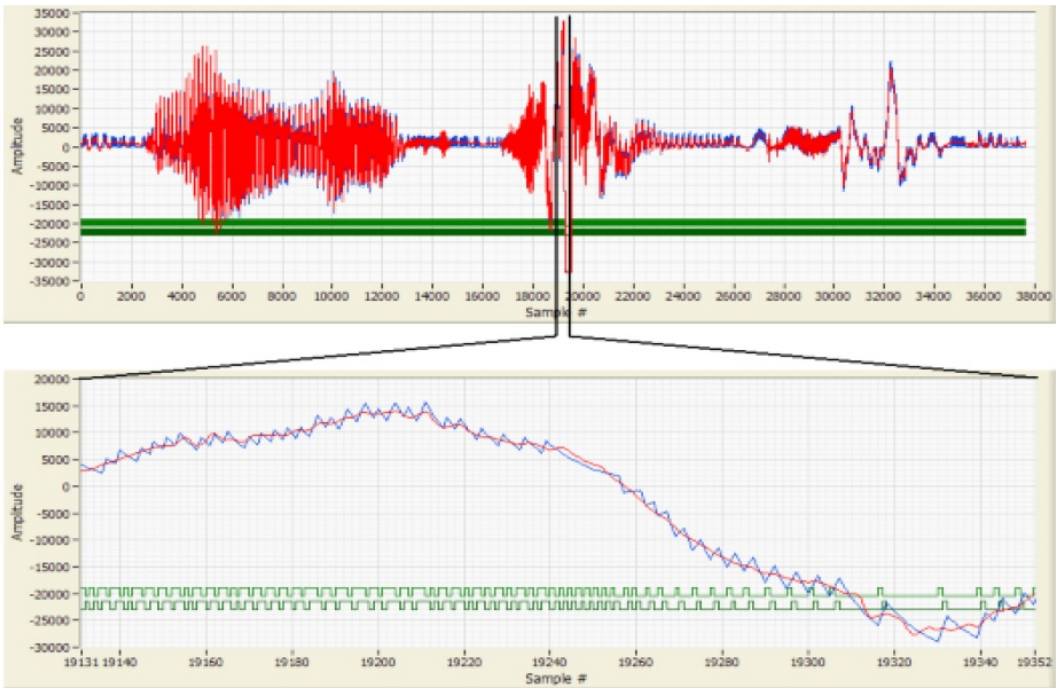

Fig. 10: Original Sound (Red Wave), Coded Sound (Blue Wave), Bit Stream (Green Wave) (Gonçalo Martins, 2008).

And also for the principle of the SVM in a-b-c coordinates is presented in (Rui Wu, Dunghua and Shaojun, 2005). The traditional SVM require a bit of digital logic and computational power to determining the duty ratio of each switch. However, the algorithm for three-phase four-leg inverter in ABC coordinates avoids the transformation, which can make the selection of the switching vectors and calculation of the duty cycle much easier, and reduce the complexity of the modulation obviously (Rui Wu, Dunghua and Shaojun, 2005). Figure (11 & 12), show the wave form simulation results.

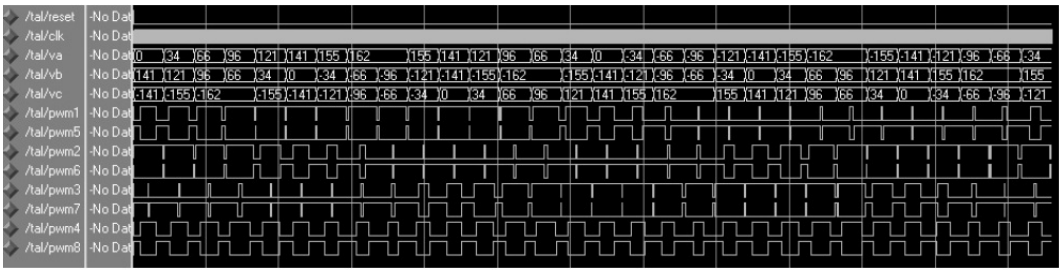

Fig. 11: Eight PMW waveform in the symmetrical reference voltage (Rui Wu, Dunghua and Shaojun, 2005).

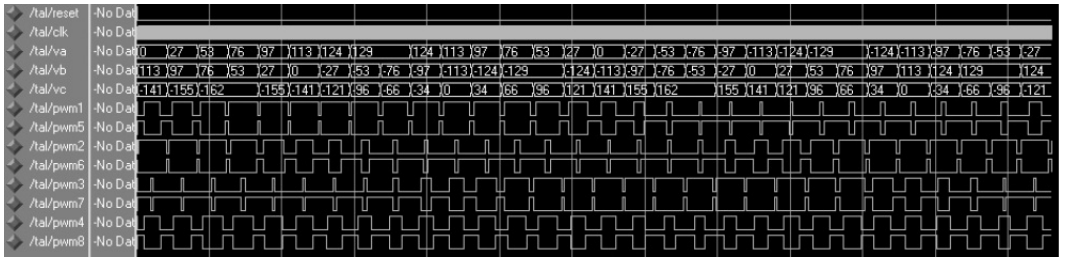

Fig. 12: Eight PMW waveform in the unsymmetrical reference voltage (Rui Wu, Dunghua and Shaojun, 2005).

The behavior is observed on different time diagrams. It is mandatory to create particular test vectors for FPU as well as for whole FzCoP. Figure (13) shows the result of FzCoP test using bit-vector and real number concepts which presented in (Radoslav Raychev, 2005).

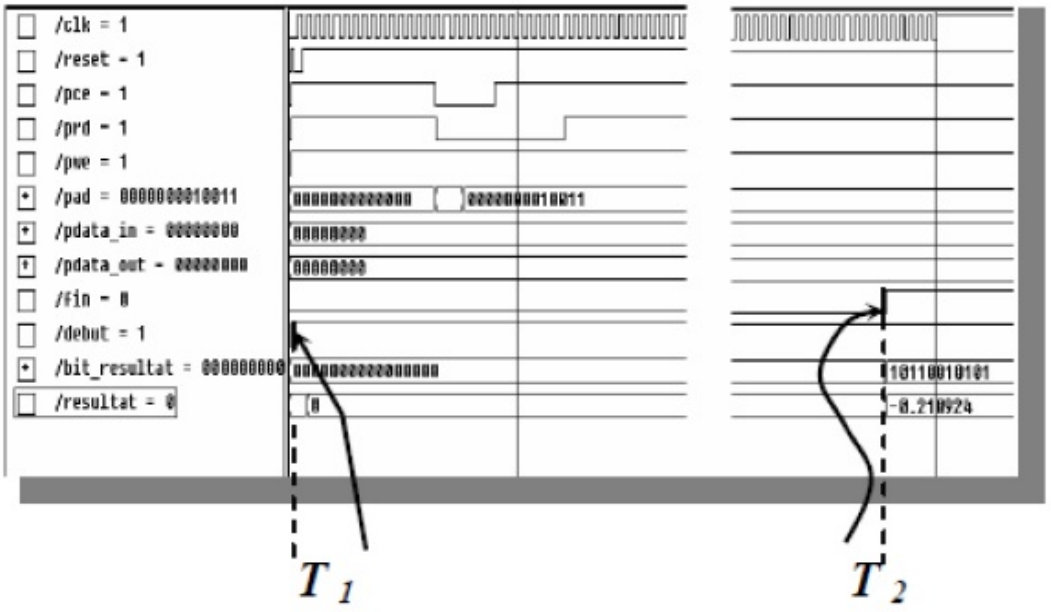

Fig. 13: An FzCoP simulation time diagram (Radoslav Raychev, 2005).

Figure (14), shows in short form, the waveform of all related signals. After a pixel P(j) has already been fetched from an image buffer at address j and assigned to the P-bus, the learning stage within neural cells for the pixel P(j) is then started. The controller supplies MAX_SUBCELL clock cycles to all neural cells (Kurdthongmee, W., 2008).

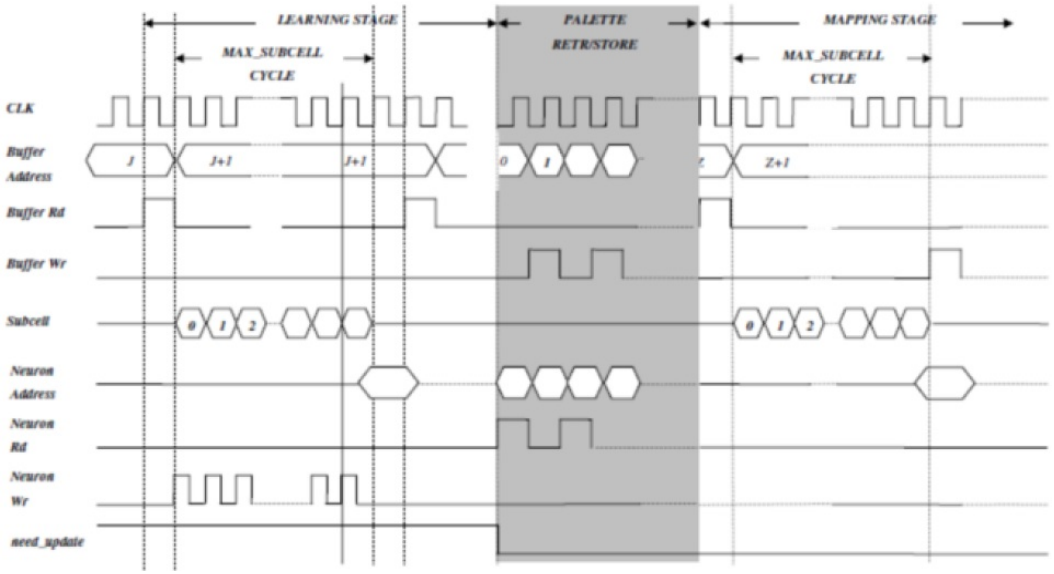

Fig. 14: The waveform of all related signals of the proposed K-SOM's controller (Kurdthongmee, W., 2008).

The offline simulator is intended to debug our microcontroller on instruction level, but allows also trace the logic levels of single signals. Due to the fact that the architecture of the microcontroller is not known a priori, the simulator has to be adaptable. For this purpose the offline simulator has distinctive simulation models of our processor cores as well as of our extension modules. These modules can be combined yielding the desired microcontroller (Martin Delvai Andreas Steininger). Figure (15) shows the performance results measured on our board for 1, 2, and 3, 4-core CerberO presented in (Antonino Tumeo Matteo, 2007), normalized with respect to the PowerPC. You can see that the execution time linearly scales with the number of cores. This proves the efficacy of our parallelization and multiprocessing platform (Antonino Tumeo Matteo, 2007).

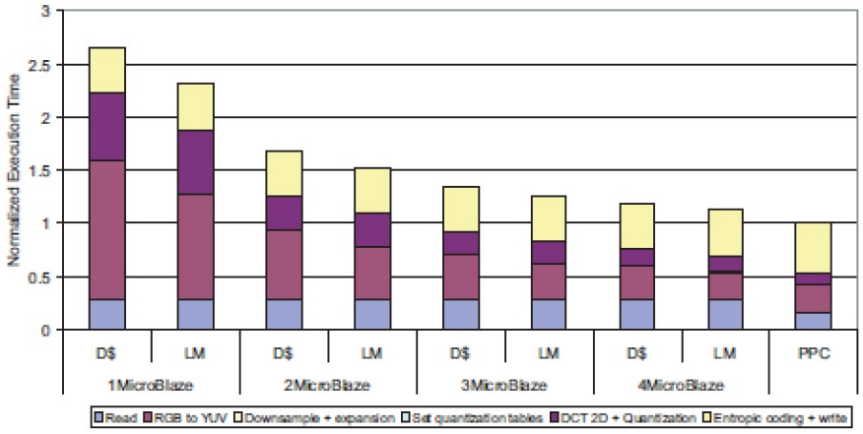

Fig. 15: Performance evaluation for CerberO w/data caches or local memories, and PowerPC (Antonino Tumeo Matteo, 2007).

The results for the first application presented in (Shuai Chey, Jie Liz,), Gaussian Elimination, are illustrated in Figure (16). For all input sizes, both FPGAs and GPUs show their advantages over CPUs, leveraging their parallel computing capabilities (Shuai Chey, Jie Liz,).

Tigers arc DSP was used in (David Rupe,) to source and sink data through the system, simulating the ADC/DAC interface, and a Stratix II FPGA acts as the bridge between the B2-AMC and the Altera Cyclone III Starter Kit (David Rupe,). Virtex II Pro FPGA speedups varied, from 37-50X the Opteron (figure 17) for 1k-16k queries with minor variations in data size are presented in (Olaf O. Storaasli, 2007).

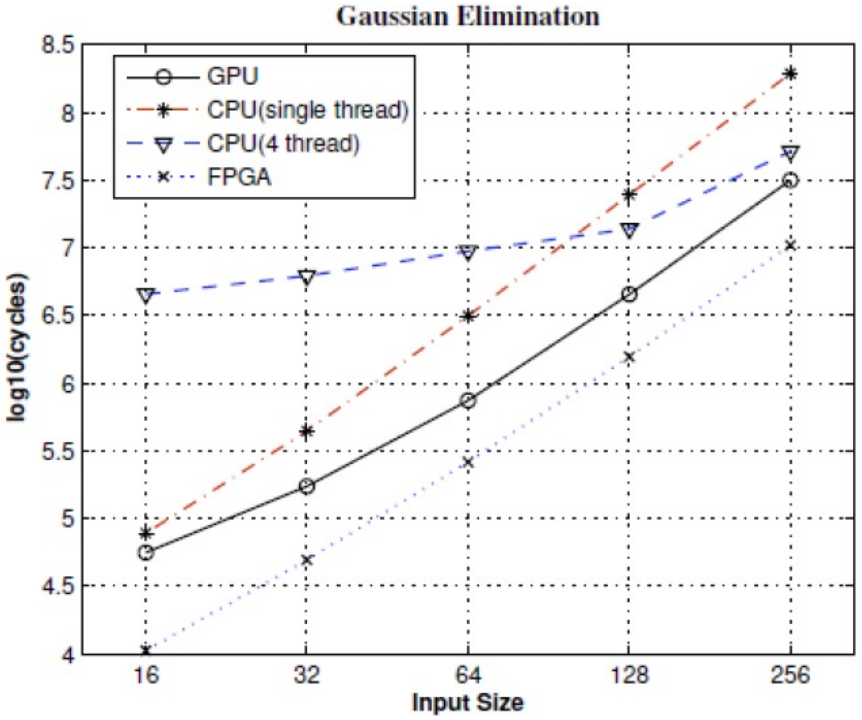

Fig. 16: Execution cycles of the four versions of Gaussian Elimination. The x-axis represents the dimensions of the computation grid. Note that the y-axis is a log scale (Shuai Chey, Jie Liz,).

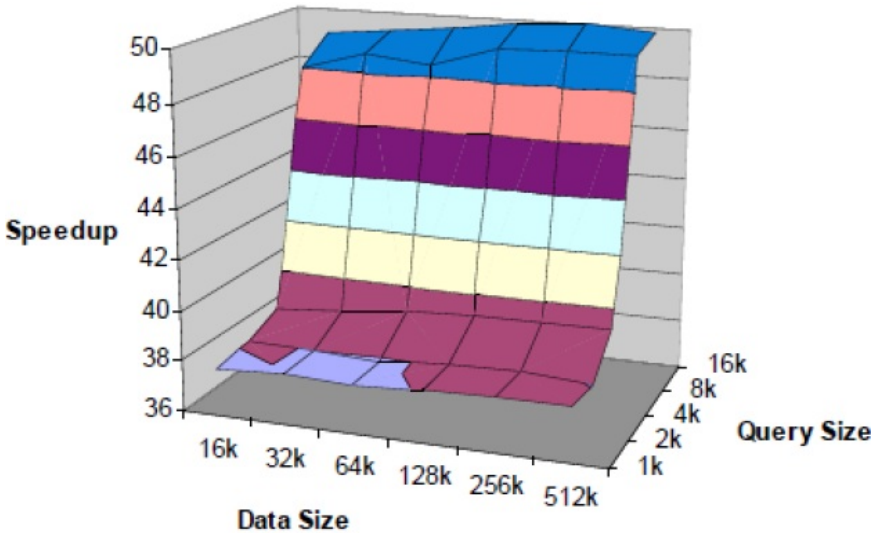

Fig. 17: Speedup for Virtex-II Pro 50 FPGA (Olaf O. Storaasli, 2007).

In Figure (18), output of high level behavioral model of BELBIC in SIMULINK and real-time embedded one (E-BELBIC) on targeted FPGA are shown. The sensory input and stress signal are fed to these models as presented in (Jamali, M.R., A. Arami, 2008).

VII. FPGA-based Implementation Results:

The techniques used for hardware implementation include: analogue implementation and digital implementation, FPGA become one of the most digital implementations used with different applications, to make use of FPGA technologies features, high speed, low coast and short time to market, in this section of our paper, we will focus on the popular implementation results which obtained from the literature. By using a high capacity of an FPGA chip, the additional hardware such as an encoder counter and a pulse width

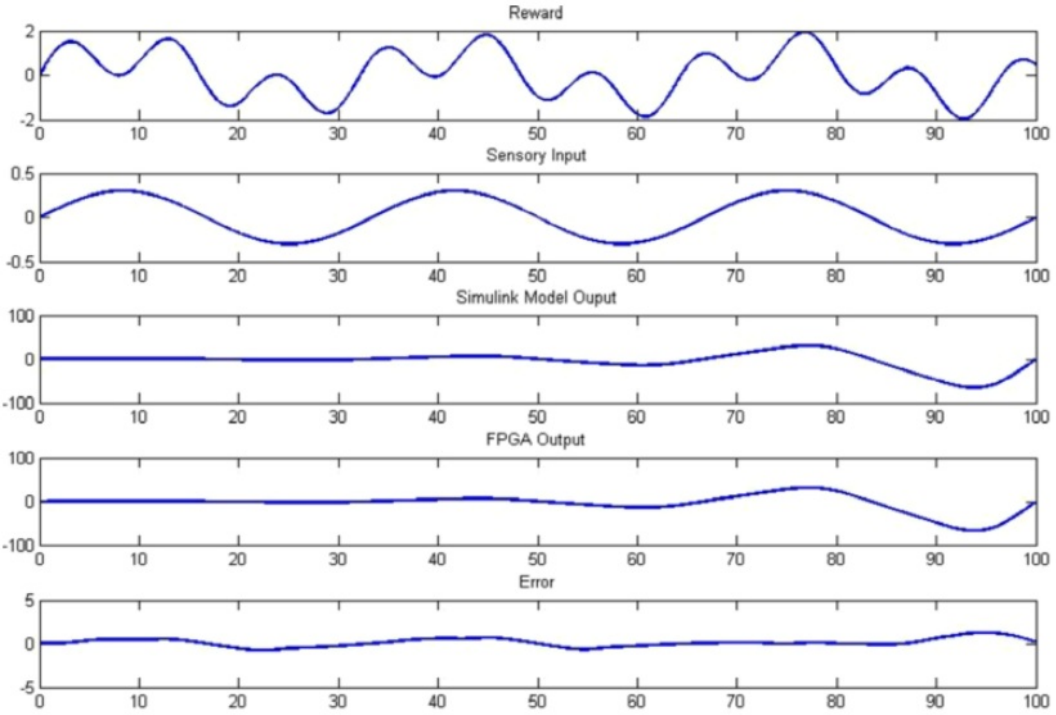

Fig. 18: High level continues SIMULINK model and targeted FPGA outputs comparison (Jamali, M.R., A. Arami, 2008).

modulation (PWM) generator is implemented in a single FPGA chip. As a result, the controller becomes cost effective. It was tested for controlling nonlinear systems such as a robot finger and an inverted pendulum on a moving cart to show performance of the controller. Experimental studies show that the implemented neural network controller works quite well for the position control of a robot finger, as well as for the inverted pendulum system (Seul Jung and Sung su Kim, 2007). A robot hand is shown in Figure (19) it has three fingers and each finger has three joints, but only two joints are actuated. DC motors with the capacity of 4.55Wof 6 V are used. The FPGA controller controls the movements of the robot hand. Movements are displayed on the screen of the computer (Seul Jung and Sung su Kim, 2007).

The hardware modules are depicted in Figure (20), the led, display and pushbutton modules were automatically generated by the EDK system. On the other hand, the keyboard module was first described in a VHDL file implementing the PS2 protocol and then included as a peripheral device in the overall design. The PWM blocks are responsible for generating modulated speed control signals of the DC-motors related to the throttle and gear devices. The PWM signals were implemented using Micro blaze's timers, which can be added to the design according to the system needs (Anderson P. Correia, 2008).

Figure (21 and 22) show the load current and flying capacitor voltage estimates from the VHDL component after D/A conversion and reverse scaling with Ki and Kv, so as to be compared to their electrical references. The estimation of the intermediate voltages is really satisfactory; since the flying capacitor voltage $x2$ and its estimate $\hat{x}2$ are really close (Figure 33) (Anne-marie lienhardt, 2007).

The first example is a standard DC-motor speed control while the other one addresses demonstration of position control of an unmanned electrical dual rotor helicopter (Mohamed Abdelati.).

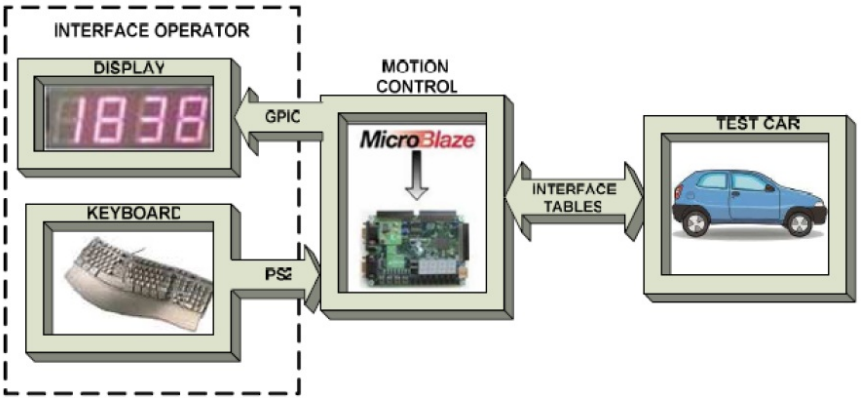

Fig. 20: Overall Hardware System (Anderson P. Correia, 2008).

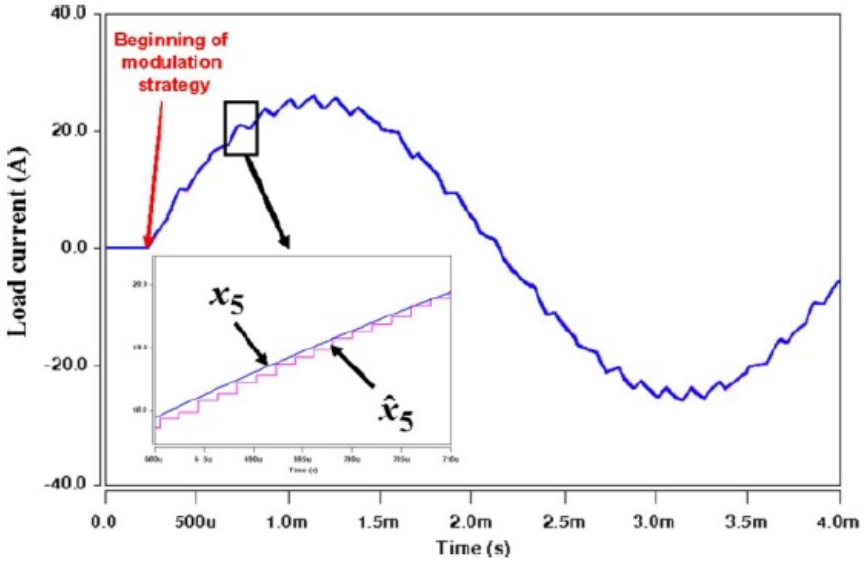

Fig. 21: Load current observation: co simulation results (Anne-marie lienhardt, 2007).

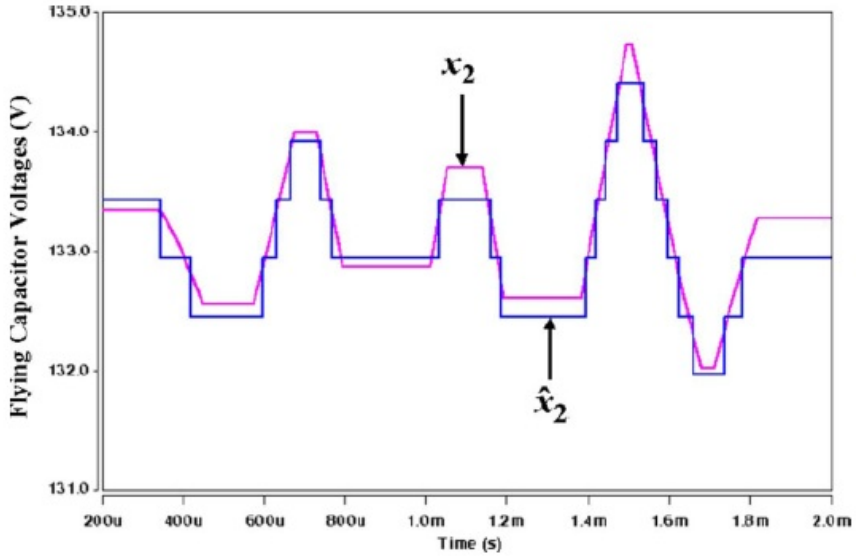

Fig. 22: Flying capacitor voltage observation: co simulation results (Anne-marie lienhardt, 2007).

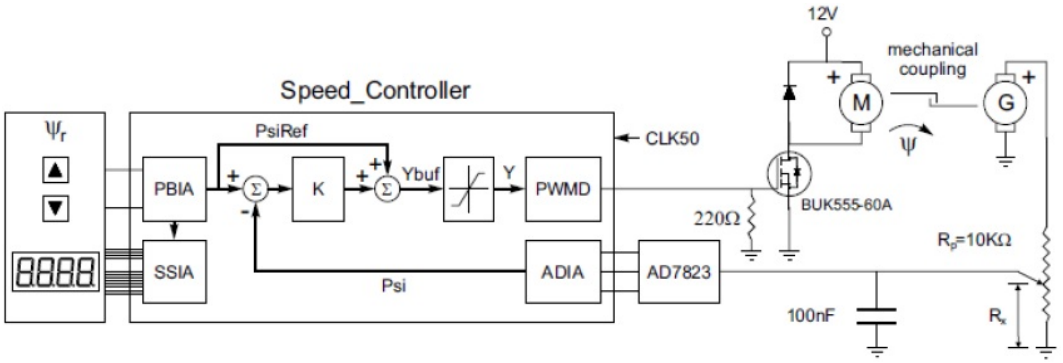

Fig. 23: DC-motor speed control using FPGA (Mohamed Abdelati.).

The experimental tests were performed with the aid of the National Instruments industrial computer with RIO PXI- 783 1R card proposed in (Jacek Lis, Czeslaw T. Kowalski, 2008), where the rotor speed is measured by the encoder for the comparison with the estimated speed only (Jacek Lis, Czeslaw T. Kowalski, 2008). Figure (24) shows the structure of the experimental system. The FPGA based driver includes an EzM-56L stepper motor, an optical encoder which generates 2500 pulses per round (10000 positions per round), and a load which is an aluminum plate. The stepper motor, encoder and load are provided by FAS Technology Co., Ltd. Proposed in (Ngoc Quy Le and Jae Wook Jeon, 2007).

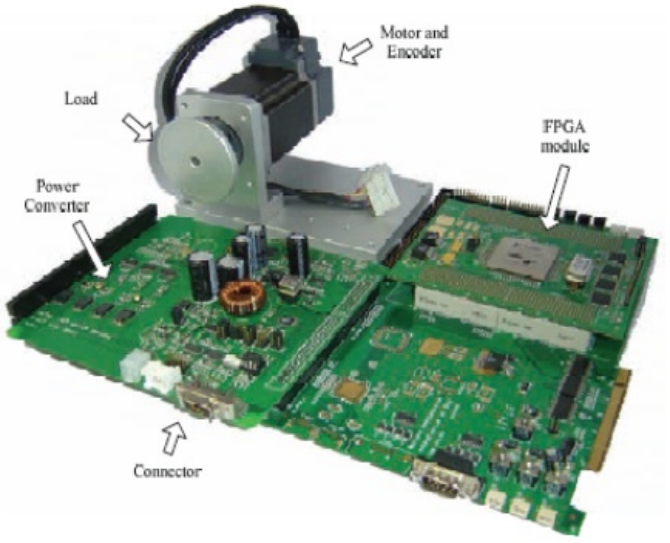

Fig. 24: Experimental system (Ngoc Quy Le and Jae Wook Jeon, 2007).

Figure (25) shows the overall system configuration of the experimental setup and the ML-300 prototyping board is the central piece with a PC and motor drive circuit attached. The board has a Virtex II pro FPGA (2VP7) on it. The GPIO pins on the board are directly connected to the FPGA device which presented in (Narashiman Chakravarthy, 2006).

Figure (26) shows the prototype boards implemented in this paper. The left board is mounted with D/A converter, connection circuit between FPGA and encoder, and peripheral circuits. The right board is an FPGA board. FPGA (Stratix EP1S10F780C7ES) made by Altera is utilized. In addition, a D/A converter (DAC813JP) manufactured by Burr–Brown is utilized, and it has +-10-V output range and 12-b resolution (Ena Ishii, 2007).

Figure (27) shows the experimental setup for the first test: Spartan-3 development board for FPGA, DAC, servo amplifier and servomotor with encoder which proposed in (Roque AlfredoOsornio-Riosa, 2009).

The deblocking algorithm produces a very noticeable improvement to the artifacts in the original image for the project presented in (Martin Hansen, 2007). Examples of input (original) images and output (deblocked) images are shown in Figure (28, a, b).

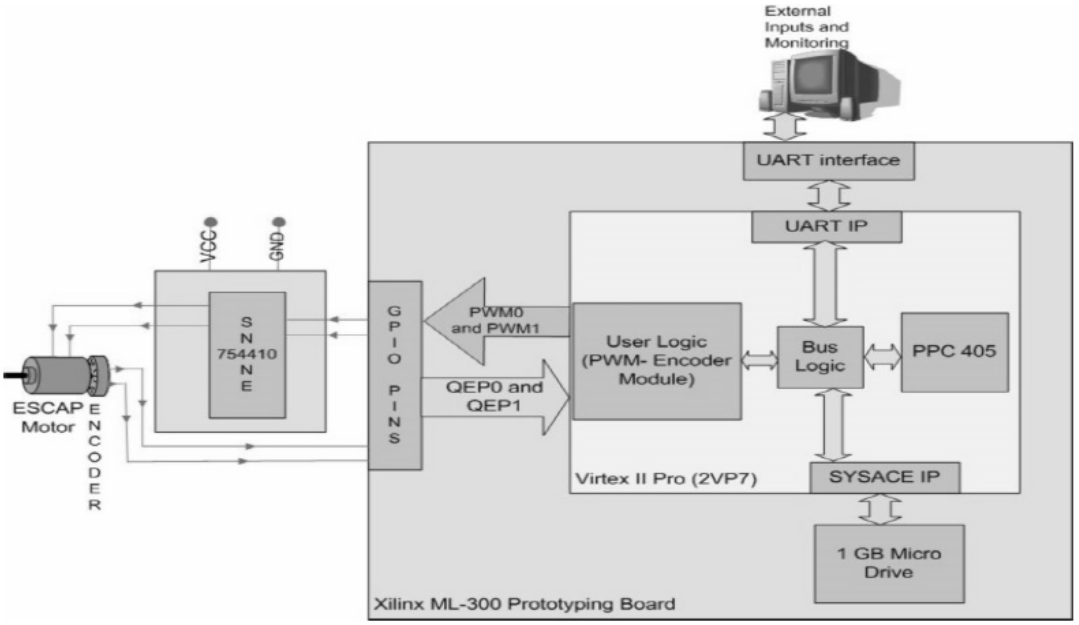

Fig. 25: Experimental Setup (Narashiman Chakravarthy, 2006).

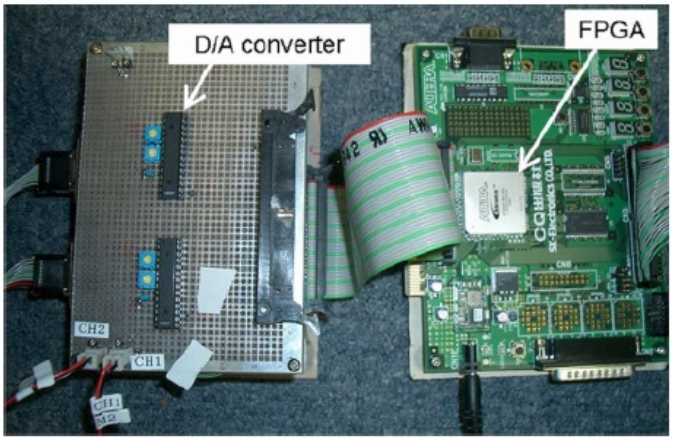

Fig. 26: Overview of implemented boards (Ena Ishii, 2007).

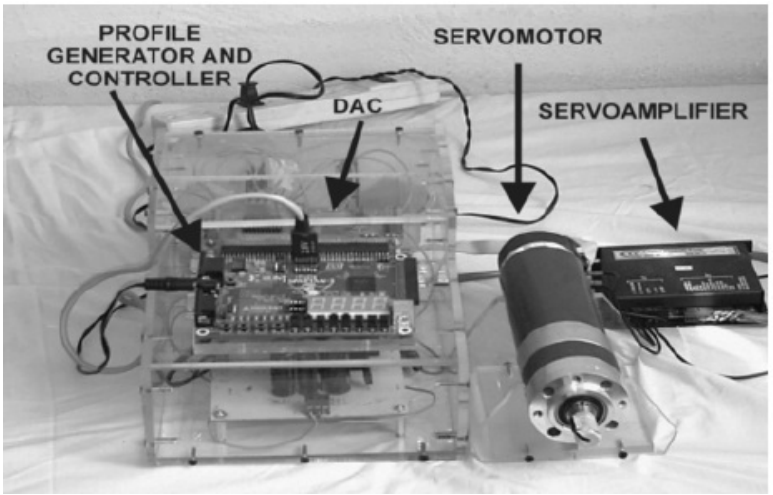

Fig. 27: Experimental setup for cases of studies 1–5 showing: profile generator, DAC, servo amplifier and servomotor (Roque AlfredoOsornio-Riosa, 2009).

(a) : Original photo.

(b) : Deblocked photo.

Fig. 28: Examples, a) Original photo, b) Deblocked photo (Martin Hansen, 2007).

The challenges and technicalities involved in implementing Simplex Algorithm for solving Integer Linear Program, on FPGA which is a Programmable Logic Device are investigated by Majumdar in (Abhinandan Majumdar, 2006). An Integer Linear Problem is basically an optimization problem, which is widely used in the area of Operations Research. It can be defined as a Linear Problem, which can take only integer values. Solving such type of problem is generally NP-hard. To show up a speed-up over software implementation is the main objective feature of the project (Abhinandan Majumdar, 2006).

VIII. Conclusion:

From this review, the authors focus and at most they preferred to implement these applications using FPGA, to make use of FPGA technology features (small device size, high speed, low coast, and short time to market). A control algorithm, when implemented in an FPGA, can have a very short execution time due to the high degree of parallelism of its architecture. At the same time, the constraints imposed by the power electronic components imply a sampling period that is, for many applications, much higher than the execution time. The resulting "wasted time" could be advantageously employed. Several examples of relevant FPGA utilizations in this context are presented in this paper. Another perspective on FPGA design is to propose a prototyping development system of a fully integrated controller from VLSI technology and SoC design that can include digital control and its analog interface (sensors, ADC, power drivers, etc.) (Eric Monmasson and

Marcian N. Cirstea, 2007). Interesting and relevant research topics were described along with the social and economic aspects of the field. Most readers would only find a subset of the topics covered relevant for their work, but the purpose was to provide a full picture in order for those readers to be better informed when constructing a threat model and corresponding defense mechanism, as a system is specified and designed (Saar Drimer, 2008). Interesting and relevant research topics were described along with the social and economic aspects of the field. Most readers would only find a subset of the topics covered relevant for their work, but the purpose was to provide a full picture in order for those readers to be better informed when constructing a threat model and corresponding defense mechanism, as a system is specified and designed (Martin Hansen, 2007). In the field of control system, many complex plants are difficult to deal with by the conventional approach because of their nonlinear, time-varying behavior and imprecise measurement information. Nevertheless, human operators can handle these complex plants by their practical experience.

ACKNOWLEDGMENT

The authors would like to thank firstly, our god, and all UPM staff and all friends who gave us any help related to this work. Finally, the most thank is to our families and to our countries which born us.

REFERENCE

André DeHon and Raphael Rubin, 2004. " Design of FPGA Interconnect for Multilevel Metallization", IEEE transactions on very large scale integration (VLSI) systems, 12(10): 1038-1050.

Abhinandan Majumdar, 2006. “FPGA Implementation of Integer Linear Programming Accelerator” Pentagram Research Centre (P) Limited., www.columbia.edu/~am2993/index_files/SS-2.2.pdf.

Anderson P. Correia, Carlos H. Llanos, Rodrigo W. De carvalho, Carla Koike, Sadek and A. Alfaro, 2008. control designs approach using a reconfigurable system for autonomous vehicle” ABCM symposium series in mechatronic - 3: 1-10 by ABCM.

Andres upegui, rico moeckel, elmar dittrich, auke ijspeert, eduardo sanchez” an FPGA dynamically reconfigurable framework for modular robotics” logic systems laboratory – http://birg2.epfl.ch/publications/fulltext/upegui05.pdf.

Antonino Tumeo Matteo Monchiero Gianluca Palermo Fabrizio Ferrandi Donatella Sciuto, 2007. A Design Kit for a Fully Working Shared Memory Multiprocessor on FPGA” ACM 978-1-59593-605-9/07/0003, pp: 219 - 222, Stresa-Lago Maggiore, Italy, March 11-13.

Anne-marie lienhardt, Guillaume Gateau and Thierry A. Meynard, 2007. Digital sliding-mode observer implementation using FPGA” IEEE transactions on industrial electronics, 54(4).

Alper Ucar and Ali Ziya Alkar. HW/SW Codesign of FPGA-based Neural Networks” Hacettepe University, Department of Electrical & Electronics Engineering, 06800 Ankara, Turkey. www.alperucar.net/tainn06_ucar_alkar.pdf.

Ali, M., Çavu_lu Cihan Karakuzu Suhap `ahin. Neural Network Hardware Implementation Using FPGA” Kocaeli University, Electronics and Telecommunication Engineering Department, Eski Gölcük Yolu Veziro lu Yerle kesi, 41040, zmit, Kocaeli ,Turkey. www.alicavuslu.com/images/Mali-cih-Suh.pdf

Aydo\an Savran and Serkan Ünsal” Hardware Implementation of A Feedforward Neural Network Using FPGAs” EGE University, Department of Electrical and Electronics Engineering. www.emo.org.tr/ekler/21eb0b827c09dd1_ek.pdf.

Andrzej Krasniewski “Application-Dependent Testing of FPGA Delay Faults” Warsaw University of Technology, IEEE explorer. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=794478

Andres Upegui, Eduardo Sanchez, “Evolving Hardware by Dynamically Reconfiguring Xilinx FPGAs” Ecole Polytechnique Fédérale de Lausanne – EPFL Logic Systems Laboratory – LSL 1015 Lausanne, Switzerland.lsl www.epfl.ch/~upegui/docs/Upegui_ICES05.pdf.

Balasaheb S. Darade and Tarun A. Parmar, 2002. Paper Presentation on Programming FPGA's Using Handel-C” Jawaharlal Nehru Engineering College Aurangabad. http://balasaheb.darade.googlepages.com/fpga.pdf

Boaz Hirschl and Leonid P. Yaroslavsky," FPGA implementations of sorters for non-linear filters", Email: boaz.hirschl@intel.com, pp: 541-544. www.eurasip.org/Proceedings/Eusipco/Eusipco2004/defevent/papers/cr1074. pdf.

Brad Alexander and Andrew Wendelborn, “A Simple Programming Model for New-Generation Hardware”, _Department of Computer Science, University of Adelaide, Adelaide 5005, Australia, January 3, 2006. www.cs.adelaide.edu.au/~brad/papers/AlexWendWOSSA06.pdf.

Brosch, O., J. Hesser, C. Hinkelbein, K. Kornmesser, T. Kuberka, A. Kugel, R. Männer, H. Singpiel, B. Vettermann, ATLANTIS – A Hybrid FPGA/RISC Based Re-configurable System, Lehrstuhl für Informatik V, Universität Mannheim, D-68131 Mannheim, YPERLINKhttp://Germany.ipdps.cc.gatech.edu/2000/raw/18000892.pdfGermany. ipdps.cc.gatech.edu/2000/raw/18000892.pdf.

Benoît R. Veillette and Gordon W. Roberts "Evaluating Dsp Systems In Real-Time Using An FPGA-Based Test Station" Microelectronics And Computer Systems (MACS) Laboratory McGill University Montréal, Québec. http://www.ece.mcgill.ca/~grober4//ROBERTS/PUBLICATIONS/CONFERENCES/1995/fpd.95/FPGA.pdf.

Dorf, R.C., 2000. "The Electrical Engineering Handbook", CRC Press LLC.

Dominic Job, Venky Shankararaman and Julian Miller, 1998. " Combining CBR and GA for Designing FPGAs" IEEE ICCIMA apos; 99. Proceedings. Third International Conference on Computational Intelligence and Multimedia Applications, 1999: 133-137.

Dr. Subbarao Wunnava and Jaime Marcelo Montenegro, 2004. " Tutorial on VHDL and Verilog Applications", Second LACCEI International Latin American and Caribbean Conference for Engineering and Technology (LACCEI'2004), Tutorial Track – Paper 023, pp: 1-6, Miami, Florida, USA.

Daryl Popig, Debra Ryle, David Pellerin and Eric Stahlberg, 2006. Applying Field Programmable Logic Arrays to Biological Problems" 3 Ohio Supercomputer Center, Columbus, OH. www.acceleratedata.com/FPGA_biometrics.pdf.

David Rupe," An FPGA Framework Supporting Software Programmable Reconfiguration And Rapid Development Of SDR Applications", (BittWare, Concord, NH, USA). www.altera.com/literature/cp/cp-01 034-spr-framework-for-development-of-sdr.pdf.

David Pellerin, 2007. "FROM 'C' to FPGA", CTO, Impulse Accelerated Technologies. http://www.abelia.no/getfile.php/ Rapporter_dokumenter/Seminarprogram/AI_FPGA_program_02.pdf.

Eric Monmasson and Marcian N. Cirstea, 2007. " FPGA Design Methodology for Industrial Control Systems-A Review. IEEE transactions on industrial electronics, 54(4): 1824-1842.

Eftichios Koutroulis, 2006. Apostolos Dollas, Kostas Kalaitzakis" High-frequency pulse width modulation implementation using FPGA and CPLD ICs" science direct, Journal of Systems Architecture, 52: 332-344.

Ena Ishii, Hiroaki Nishi and Kouhei Ohnishi, 2007. "Improvement of Performances in Bilateral Teleoperation by Using FPGA" IEEE transactions on industrial electronics, 54(4): 1876-1884, august.

FPGA tutorial "Over view on FPGA", www.Tutorial-reports.com, 2008. http://www.tutorial-reports.com/computer-science/fpga/overview.php

Franjo Plavec, Zvonko Vranesic and Stephen Brown "University of Toronto Department of Electrical and Computer Engineering 10 King's College Road, Toronto, Ontario, Canada.www.eecg.toronto.edu/wosps08/papers/plavec.pdf.

Gonçalo Martins, Manuel Barata, Luís Gomes, 2008. "Low Cost Method to Reproduce Sound with FPGA" IEEE International Symposium on Industrial Electronics, ISBN: 978-1-4244-1665-3, pp: 1932-1936, Cambridge.

Grout, I.A. and K. Keane. A Matlab to VHDL Conversion Toolbox for Digital Control" Department of Electronic and Computer Engineering, University of Limerick, Limerick, Ireland. www.ece.ul.ie/homepage/ian_grout/paper1.pdf.

Hwu, K.I., Y.T. Tau, 2005. A forward converter having a FPGAs-based PID controller with parameters on- line tuned", IEEE International Conference on Power Electronics and Drives Systems, 2: 1239- 1243.

Hans-Peter Röser and Felix Huber, BOSS: Software and FPGA Middleware for the "flying laptop" Microsatellite" **Universität Stuttgart Institut für Raumfahrtsysteme Pfaffenwaldring 31 70550 Stuttgart.sergio.montenegros.de/public/montenegro_boss-flyinglaptop.pdf.

Jamali, M.R., A. Arami, M. Dehyadegari, C. Lucas, Z. Navabi, 2008. Emotion on FPGA: Model driven approach" ELSEVIER, Expert Systems with Applications, 0957- 4174, pp: 1-10.

Jingeng Mai, Qining Wang, and Long Wang, FPGA-Based Gait Control System for Passive Bipedal Robots" National Natural Science Foundation of China (No. 60404001 and No. 60274001).planning.cs.cmu.edu/humanoids07/p/28.pdf.

Jacek Lis, Czeslaw T. Kowalski, Teresa Orlowska-Kowalska, 2008. "Sensor less DTC Control of the Induction Motor using FPGA" IEEE International Symposium on Industrial Electronics, ISBN: 978-1-4244-1665-3, pp: 1914 - 1919, Cambridge.

Joy, S.P. Vasantha Rani P. Kanagasabapathy. Design of Neural Network on FPGA" Madras Institute of Technology, Anna University Chennai-600 044,Tamil Nadu, India. http://68.153.197.235/CSREA/VLS2028.pdf

Jingzhao Ou and Viktor K. Prasanna. A Methodology for Energy Efficient Application Synthesis Using Platform FPGAs" This work is supported by United States National Science Foundation (NSF) under award no. CCR-03 11823, pp: 1-4. ceng.usc.edu/~prasanna/papers/ouj_ersa04.pdf.

Jason Villarreal, John Cortes and Walid A. Najjar, 2007. “Compiled Code Acceleration of NAMD on FPGAs” Jason Villarreal, Department of Computer Science and Engineering University of California. www.cs.ucr.edu/~najjar/papers/2007/RSSI2007.pdf.

Jean Pierre Banˆatre, Dominique Lavenier and Marc Vieillot, 1994. “From High Level Programming Model to FPGA Machines” Institut National De Recherche En Informatique Et Automatique, ISSN 0249-6399, BP 105, 78153 LE CHESNAY Cedex (France).

Kurdthongmee, W., 2008. A novel hardware-oriented Kohonen SOM image compression algorithm and its FPGA implementation” Science Direct, Journal of Systems Architecture, 54: 983-994.

Kai Wang , Van Yuan , Qianyi Wang Tomoo Aoyama “One-chip FPGA Implementation of Multi-Valued "And/Or"-Neural Network” Research student, Department of Electric and Electronics Engineering (DEEE), pp: 175- 181. http://ir.lib.miyazakiu.ac.jp/dspace/bitstream/123456789/262/3/KJ00002425534.pdf.

Lieu My Chuong, Lam Siew Kei, Thambipillai Srikanthan, 2008. High-Level Delay Estimation Technique for Porting C-based Applications on FPGA” IEEE International Symposium on Industrial Electronics, ISBN: 978-1-4244-1665-3, pp: 1991-1997, Cambridge.

Lee W. Howes, Paul Price, Oskar Mencer and Olav Beckmann, 2007. “FPGAs, GPUs and the PS2 - A Single Programming Methodology “Department of Computing, Imperial College London.

Maxim Leonov and Vyacheslav Kitaev,” Multi-Channel Correlation of Wideband RF Signals In Hybrid CPU+FPGA System” Engineering Research Institute Auckland University of Technology Private Bag 92006, Auckland 1142, New Zealand. http://crs.aut.ac.nz/publications.

Maxfield, C., 2004. "The Design Warrior's Guide to FPGAs", Mentor Graphic's corporation and Xilinx Inc., USA.

Mohamed Abdelati. FPGA-Based PID Controller Implementation” This research was supported by the Ministry of Higher Education in Palestine, The Islamic University of Gaza. http://www.iugaza.edu.ps/ara/research/articles/volume%2014-%20Issue%201%20-studies%20-7.pdf.

Miroslav Vadkerti, 2006. JTAG programming device for FPGA configuration EEPROMs” CESNET technical report number 29/2006. www.cesnet.cz/doc//techzpravy/2006/jtag/jtag.ps.gz.

Martin Delvai Andreas Steininger” Teaching Hardware Software Co-Design to Software Engineers” Vienna University of Technology Institute of Computer Engineering Vienna, Austria.xputers.informatik.uni-kl.de/RCeducation06/RCe-29.pdf.

Martin Hansen, 2007. Implementing Real-Time Video Deblocking in FPGA Hardware” A thesis presented to the University of Waterloo, Waterloo, Ontario, Canada. http://uwspace.uwaterloo.ca/bitstream/10012/2990/1/ThesisMain.pdf.

Naylor, G.A., 2003. ESRF Grenoble France “Fast DSP Using FPGAS And Dsos For Machine Diagnostics” Proceedings DIPAC – Mainz, Germany, Contributed Talks CT07, pp: 71-73.

Ngoc Quy Le and Jae Wook Jeon, 2007. " An Open-loop Stepper Motor Driver Based on FPGA", International Conference on Control, Automation and Systems, Seoul, Korea, 978-89-950038-6-2-98560/07, pp: 1322-1326.

Narashiman Chakravarthy, Jizhong Xiao, 2006. FPGA-based Control System for Miniature Robots”, IEEE International Conference on Intelligent Robots and Systems, ISBN: 1-4244-0258-1, pp: 3399-3404, Beijing, China, October 9-15.

National, Instruments, 2007. “Introduction to FPGA Technology: Top Five Benefits”, www.ni.com. http://eee.guc.edu.eg/Courses%202008%20WS/ELCT/ELCT902%20Programmable%20Logic%20Circuits/Reading%20Material/tut_6984_R2.pdf.

Olaf O. Storaasli, 2007. “Exploring Accelerating Science Applications with FPGAs” Oak Ridge National Laboratory, Cray Inc, rssi.ncsa.uiuc.edu/proceedings/papers/rssi07_08_paper.pdf.

Oskar Mencer, Marco Platzner, Martin Morf and Michael J. Flynn, 2001. “Object-Oriented Domain Speci_c Compilers for Programming FPGAs” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 9(1): 205-210.

Rob Baxter1, Stephen Booth, Andrew McCormick and Allan Cantle, 2007. “Maxwell – a 64 FPGA Supercomputer “University of Edinburgh, James Clerk Maxwell Building, King’s Buildings, Edinburgh EH9 3JZ, www.fhpca.org/download/RSSI07-Maxwell.pdf.

Roman Kohut and Bernd Steinbach, 2003. “The Structure of Boolean Neuron for the Optimal Mapping to FPGAs” Bernhard-von-Cotta-Straße 2, D-09596 Freiberg, GERMANY. www.informatik.tu-freiberg.de/prof2/publikationen/CADSM2005_BN_FPGA.pdf.

Radoslav Raychev, Abdellatif Mtibaa, Mohamed Abid, 2005. " VHDL Modelling of a Fuzzy Co-processor Architecture", International Conference on Computer Systems and Technologies – CompSysTech, pp: I.2-1-I.2-6.

Rui Wu, Dunghua and Shaojun, 2005. "A three dimensional space vector modulation algorithm in A-B-C coordinate implemented by a FPGA" 31st Annual Conference of IEEE, ISBN: 0-7803-9252-3, pp: 1071 – 1075, Raleigh, NC.

Roque AlfredoOsornio-Riosa, Rene de Jesu Romero-Troncoso, 2009. FPGA implementation of higher degree polynomial acceleration profiles for peak jerk reduction in servomotors" science direct, Robotics and Computer-Integrated Manufacturing, 29: 379-392.

Richard Wain, Ian Bush, Martyn Guest, Miles Deegan, Igor Kozin and Christine Kitchen, 2006. An overview of FPGAs and FPGA programming; Initial experiences at Daresbury", Daresbury, Warrington, Cheshire, WA4 4AD, UK, November 2006 Version 2.0. www.cse.scitech.ac.uk/disco/publications/ FPGA_overview.pdf.

Robert Trausmuth, 2006. Christian Dusek and Using Faust For FPGA Programming "Proc. of the 9th Int. Conference on Digital Audio Effects (DAFx-06), Montreal, Canada, September 18-20, DAFX-287- DAFX-290.

Rebaudengo, M., M. Sonza Reorda, M. Violante. A new functional fault model for FPGA Application-Oriented testing" This work has been partially supported by Center of Excellence on Multimedia Radiocommunications (CERCOM) of Politecnico di Torino, Politecnico di Torino Torino, Italy. www.cercom.polito.it/Publication/Pdf/150.pdf.

Reetinder P.S. Sidhu, Alessandro Mei and Viktor K. Prasanna, 1999. Genetic Programming using Self-Reconfigurable FPGAs" 9th International Workshop on Field Programmable Logic and Applications, August.

Sornam V. Viswanathan, 2005. "Embedded Control Using FPGA", Seminar Report, Interdisciplinary Programme in Systems and Control Engineering, Indian Institute of Technology, Bombay Mumbai November, www.sc.iitb.ac.in/~sornam/seminar.pdf.

Seul Jung and Sung su Kim, 2007. Hardware Implementation of a Real-Time Neural Network Controller With a DSP and an FPGA for Nonlinear Systems" IEEE transactions on industrial electronics, 54(1): 265- 271 February.

Scott Hauck, 1998. The Roles of FPGAs in Reprogrammable Systems" Proceedings of the IEEE, 86(4): 615-639, April.

Saar Drimer, 2008. Volatile FPGA design security - a survey" The author is supported by a grant from Xilinx, Inc, Version 0.96, April 17. www.cl.cam.ac.uk/~sd410/papers/fpga_security.pdf.

Speers1, T., J.J. Wang1, B. Cronquist1, J. McCollum1, H. Tseng1, R. Katz and I. Kleyner, 0.25 m FLASH Memory Based FPGA for Space Applications" speers, B6. www.actel.com/documents/FlashSpaceApps.pdf.

Shuai Chey, Jie Liz, Jeremy W. Sheaffery, Kevin Skadrony and John Lachz "Accelerating Compute-Intensive Applications with GPUs and FPGAs "Kevin Skadron is currently on sabbatical with NVIDIA Research, University of Virginia. www.cs.virginia.edu/~skadron/Papers/che_sasp08.pdf.

Sunghyun Lee, Kiwook Yun, Kiyoung Choi, Seongsoo Hong, Soomook Moon and Jeonga Lee, 2001. Java-Based Programmable Networked Embedded System Architecture with Multiple Application Support" Department of Computer Science, Chosun Univ, 2001. http://poppy.snu.ac.kr/papers/icda2000.pdf.

Simon Bevan, A. "lecture on Programming the FPGA" page 65-8 1, www.hep.ucl.ac.uk/~ dwaters/3C41/Chapter7.pdf.

Steve Trimberger, 2007. "Trusted Design in FPGAs" ACM 978-1-59593-627-1/07/0006, June 4–8, 2007, San Diego, California, USA, DAC.

Tom Van Court and Martin C. Herbordt "Requirements for any HPC/FPGA Application Development Tool Flow" the NIH through award #RR020209-01 and facilitated by donations from Xilinx Corpo-ration. Web: http://www.bu.edu/caadlab.

Thomas Wollinger and Christof Paar, 2003. "How Secure Are FPGAs in Cryptographic Applications? (Long Version)" This research was partially sponsored by the German Federal Office for Information Security (BSI), Ruhr-Universit¨at Bochum, Germany, http://eprint.iacr.org/2003/119.pdf.

Tom Kean, 2002. "Cryptographic Rights Management of FPGA Intellectual Property Cores" FPGA 2002, February 24-26. Monterey, CA, USA, ACM 1-58113-452-5/02/0002.

Uffe Jakobsen and Torben Matzen, 2008. Design of a dedicated processor for ac motor control implemented in a low cost FPGA" IEEE Power Electronics Specialists Conference, ISBN: 978-1-4244-1667-7, pp: 3048 - 3053.

Viejo, J., J. Juan, M.J. Bellido, E. Ostua, A. Millan, P. Ruiz-de-Clavijo, A. Mu˜noz and D. Guerrero, 2008. Design and Implementation of a SNTP Client on FPGA" IEEE International Symposium on Industrial Electronics, ISBN: 9781424416660, pp: 1971-1975, Cambridge.

Vince Edwards, Michael Courtney, Kuo-pao Yang, 2008. "A FPGA Paint Brush Application" Southeastern Louisiana University Hammond, Louisiana 70402, USA, Proc ISECON, 25: 1-7, (Phoenix).

Wutthikorn Threevithayanon, Kittiphan Techakittiroj, Narong Aphiratsakun, and Soe moe Nyun, 2005. "XC2 S50 FPGA Board for Microprocessor and Digital Design Applications" Bangkok, Thailand, AU J.T., 9(1): 41-45.

Xiaodong Zhang, Junjun He and Chunlei Sheng, 2005. An approach of micro-stepping control for the step motors based on FPGA" IEEE International Conference on Industrial Technology, ISBN: 0-7803-9484-4, pp: 125 - 130, Hong Kong.

Yaodong, Tao, Hu lin, Yi hu~ xiaohui zhang~ zhicheng wang, 2008. efficient implementation of CNC position controller using FPGA" The IEEE international conference on industrial informatics (INDIN 2008), ISSN: 1935-4576, pp: 1177- 1182, DCC, daejeon, korea july 13-16.

Yasuo Sakaide, Kenichi Chiba, Yoshiya Iide, Kenji sugimoto, 1999. "Practical Test Method of Programmed Antifuse FPGA For Space Applications" Daimon, Minato-ku, TOKYO, 105-0012. www.klabs.org/mapld04/papers/p/p149_sakaide_p.doc.