Development of Integral Environment in Matlab/Simulink for FPGA

Dejan JOKIC[1], *Slobodan LABURA*[1], *Stevan STANKOVSKI*[2]

[1]Faculty of Electrical Engineering, University of East Sarajevo, Vuka Karadzica 30, 71126 Lukavica, East Sarajevo, Bosnia and Herzegovina
[2]Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja c 6, 21000 Novi Sad, Serbia

dejan.jokic@etf.unssa.rs.ba, slobodan.lubura@ etf.unssa.rs.ba, stevan@uns.ac.rs

Abstract. *In this paper is presented realization of integral environment which consists of software and hardware components for the purpose of programming Altera DE boards. Software component is Toolbox FPGA Real Time which enables simple use of Matlab/Simulink with DSP Builder for the purpose of realization of control structures. Hardware component are Interface cards that make connection of DE board with object of control possible. Simulation and experimental results of DC motor control indicate the usefulness of the proposed concept.*

Keywords

DSP Builder, Embedded systems, FPGA, Matlab/Simulink.

1. Introduction

The progress in FPGA circuits is contributed to rapid development of CAD tools and design methodology of these circuits in order to switch from traditional design methodology (HDL) to abstract methods of design [1]. This means that engineers of different vocations who are not familiar with internal structure of FPGA circuits and HDL language may program these circuits with simplicity. In this manner manufacturers of FPGA circuits offered specialized software tools in the market such as DSP Builder (Altera) to overcome this conceptual gap. DSP Builder is a component of widely used and accepted software package Matlab/Simulink and is used for digital design entry, import of digital design realised in traditional manner using Quartus II or any other HDL language and programming of the FPGA circuits. As Matlab/ Simulink is widely used for development and validation of control structures, intuitive idea was born that those control structures and its components (PID controller, encoder interface block, PWM generator block, etc.) could also be realised using Matlab/DSP Builder [1], [2]. In such way, prototypes of the control structures would be simply derived and, if necessary, implemented for certain application as ASIC circuit. Also, the advantage of using this design method is the possibility to connect digital circuits derived from DSP Builder with other blocks in Matlab environment for simulation purposes. However, DSP Builder, unlike Simulink, has only basic blocks making design of control structures on FPGA circuit very difficult and demanding. This drawback is recognized as main research issue and it is the main focus in scope of research in this paper. The realization of toolbox with blocks for describing complex functions is providing opportunity for engineers from different areas to simply develop control algorithms on FPGA circuits without any deeper knowledge in internal structure of FPGA circuit or some experience in HDL programming.

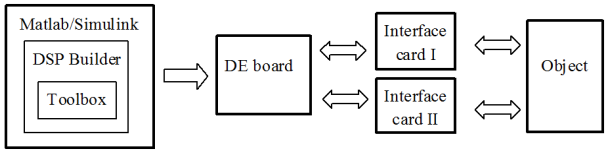

Fig. 1: Block diagram of integral environment for design control structures.

2. Components of Integral Enviroment

The integral environment has software (Toolbox) and hardware (Interface card) component offering simple method for design of control algorithms without engaging in specificity of programming FPGA circuit. In Fig. 1 is shown integrated environment for design control structure in Matlab/DSP Builder. The derived toolbox is a part of Matlab/Simulink environment and

is named FPGA Real Time. This FPGA Real Time toolbox can be used for design of control structures on all Altera FPGA DE development boards. Connections between DE board and object of control are realised with interface board. Created environ-ment can be used for lab exercises within variety of courses:

- robotics and mechatronics,
- power electronics,
- difital electronics

and other subjects which deal with control tasks.

3. Custom Toolbox

Main aim of research presented in this paper was development of integral environment, namely development of toolbox in Matlab/DSP Builder for design and testing one broad class of control algorithms on FPGA platforms. The previous methods for design of control structure of FPGA circuit were relying on conventional digital circuits design based on HDL language description. It was unknown for many engineers who were used to comfortable methods for design such as Matlab, LabView etc. with graphical environment. The new software package DSP Builder is software with the purpose to, at least partially, overcome this conceptual gap between high-level graphics programming and HDL low-level design methods. In order to use Matlab for design of control structures on FPGA circuits it is necessary to install software tool DSP Builder as part of Matlab/Simulink environment. DSP Builder has basic blocks for hardware description on FPGA circuit (basic logic circuit, flip-flop, delay circuit, etc.) but they are not sufficient for design of control structure and it was necessary to deploy required blocks in order to simplify design control structures on FPGA circuit. In Fig. 2. is shown deployed toolbox named FPGA Real Time. Only main blocks are described in this paper.

During design phase, a special attention has been devoted to deployed blocks regarding a wide area of use of these blocks. The most widely used blocks and their purpose will be descri-bed hereafter. FPGA Real Time has following blocks:

- PID controller,
- assignment values of parameters of PID controller,
- encoder interface,
- velocity estimator,
- output register (zero order hold),

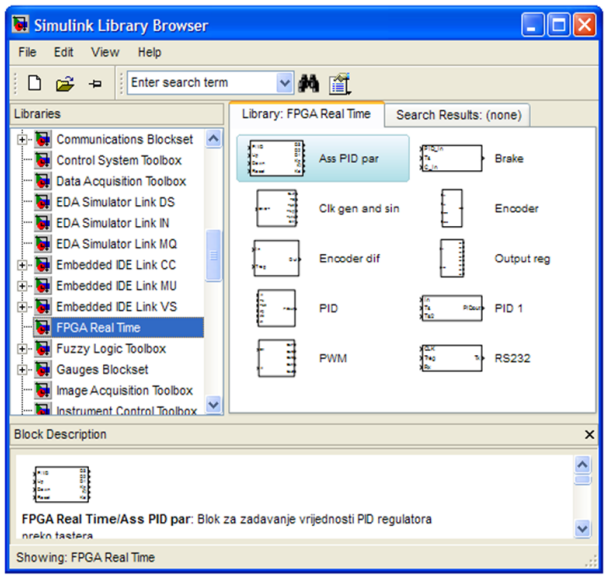

Fig. 2: Deployed FPGA real time toolbox in matlab/ DSP builder environment.

- clock generator and sine wave generator,
- PWM generator,
- RS 232.

The deployed toolbox can be used in software package Matlab/Simulink environment with installed DSP Builder toolbox in the same way and without any limitations as other toolboxes are used.

3.1. PID Controller Block

The process quantities often used in practice (temperature, pressure, level, flow, etc.) must accomplish default values in both states, in the stationary state (static accuracy) and in dynamic transient state (desired response). Today, the most common industry controllers used are PID (proportional–integral–derivative) controllers. Previously mentioned regulator is quite simple for implementation and requires modest processor resources. The digital form of this controller is suitable for implementation on FPGA circuits. Digital form of PID controller can be described as [1]:

$$\tau_{PID}(z) = \left(K_p + K_i \frac{Ts}{2} \frac{z+1}{z-1} + K_p \frac{z-1}{z}\right) E(z). \quad (1)$$

Integral part of PID controller is accomplished by Tustin approximation that gives better results than Euler's forward and backward approximation [1]. Open-loop response of deployed PID controller in Matlab environment for unit step signal at input has been tested by simulation. Figure 3 shows responses of two PID controllers. The one is deployed with standard

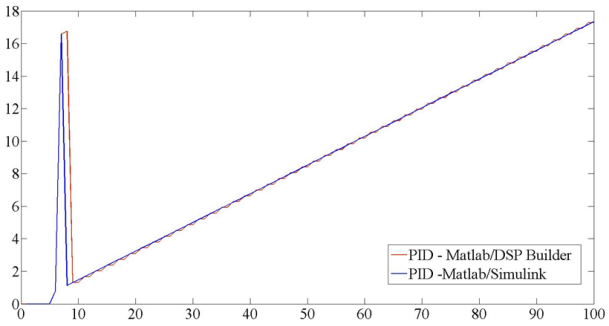

Fig. 3: Open-loop response of PID controller deployed in Matlab and DSP Builder environment.

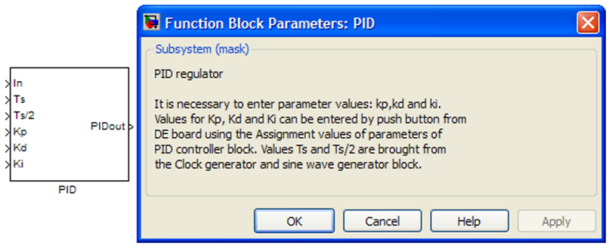

Fig. 4: Block PID controller realized in DSP Builder toolbox.

Matlab blocks and the other by using DSP Builder. The abscissa axis presents the time specified by number of clock periods Ts where duration of one period is 1.3 ms. The chosen clock period of 1.3 ms is commonly used for design of various control structures such as a motor drive, robotics, mechatronics, etc.

Analysing the responses of both PID controllers, it is possible to conclude that PID controller realized in DSP Builder has similar responses as PID controller realized with Matlab blocks, therewith that PID controller realized in DSP Builder generates output signal each $Ts = 1.3$ ms causing certain mismatch between their responses. On the Fig. 4 is shown PID controller block that has inputs for entry parameters of PID controller (K_P, K_i and K_d).

The idea is that values of parameters can be manually entered by buttons on FPGA DE development board in real time. In this way, there is possibility to obtain responses of designed control structure immediately without need for compiling FPGA circuit again.

3.2. Encoder Interface

In order to design position control structures that are common in robotics, mechatronics, etc., it is necessary to have information about real position of the controlled object. The most common component for this purpose is an incremental encoder. Incremental optical encoders are used to generate two pulses, A and B used for calculating the corresponding angular displacement. They have a source of light and light

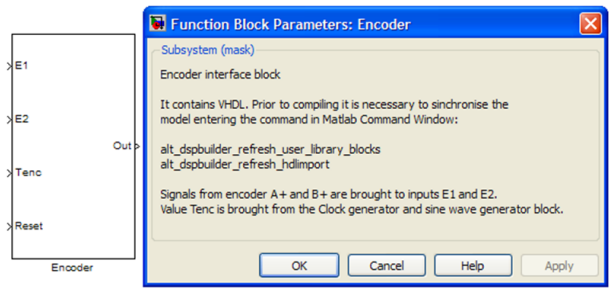

Fig. 5: Encoder interface block.

receiver where a disc with openings is positioned between them. The number of openings is proportional to the number of impulses (encoder resolution), so by counting the impulses the information about angular displacement can be acquired. At encoder output there are two symmetrical pulses A and B which phase 90° and index output Z. Turning direction is determined by which pulse, A or B is ahead in relation to one another. Incremental encoders are produced in wide resolution spectrums, from 500 to 5 000 impulses. Encoder outputs support various logic standards and interfaces such as HTTL, TTL, RS485, RS422 etc. and those are encoders with embedded electronics. There are also encoders which provide unprocessed signal, in other words sinus, usually with 1 VPP voltage at output, and they are not supported by interface cards which means that those signals have to be additionally processed [3]. The interface block design (Fig. 5) serves for the purpose of accepting and processing the signal from encoder. Outputs of encoder A and B are connected to inputs E1 and E2. Clock pulses for noise filtering of inputs A and B are connected to the input T_{enc}. Signals that are accepted from encoder output may contain noise and jitters. Their presence has negative effect on correct counting of impulses that as a result provides incorrect information about the position.

In order to determine minimum duration of impulses t under which every short impulse is considered to be jitter, it is necessary to define the largest number of impulses that are generated by the encoder. This number of impulses depends on encoder resolution and motor rotating speed. For example, we can say that encoder resolution is 1 000 impulses and motor rotation speed 3 000 rpm. In that case total number of impulses generated in one minute is 3 000 000, or 50 000 imp/sec. It follows that minimal duration of a single encoder impulse is $T = 20$ µs. Considering the fact that duty cycle is 50 %, the shortest impulse lasts 10 µs, so it is guaranteed that every impulse shorter than 2.5 µs can be considered to be a jitter (which was confirmed experimentally [4], [5]. After filtering the pulses we need to fourfold multiply the resolution using finite state machine (FSM). By observing rising and falling impulse edges of channels A and B it is possible to generate

new pulse AB whose number of impulses is four times greater than input impulse number of channels A and B. Inverting channels A and B we get four pulses where on every rising edge one impulse in new pulse AB is generated. In such manner we ensure that counter values on encoder signal accepting block output match the measured position of incremental encoder. For the purpose of incrementing and decrementing of counter values it is necessary to get the information about encoder disc rotation direction to counter control input. That information is also generated by FSM described in VHDL language (Quad_decoder [6], [7]) and it represents the main part of the design (Fig. 6) which is in charge for processing encoder signal. Design for encoder signal accepting and processing is provided in Fig. 6. where two filters for jitters with inputs E1 and E2 were described. Their outputs are brought to Quad_decoder block input which provides processed pulse Count and information about rotation direction Up/down at its output and both information are led to Up/Down counter (Counter). At counter output there is bus formation block with 16-bit width. Encoder signal accepting and processing design verification was first performed in Matlab (Fig. 7). The design first had to be able to correctly accept 1000 ppr encoder signal at speed of 3000 rpm, which corresponds to characteristics of encoder used for design functionality experimental verification. Simulation results in Fig. 7. show four signals. On the x axis is simulation time which emulates FPGA circuit global clock.

FPGA circuit global clock period is 20 ns which cannot be achieved because of computer restrictions so for the simulation purposes we used the shortest amount of time available which was 75 ns (total simulation time is 75 µs = 1000·75 ns. The first signal (marked in black) is clock signal Tenc which is brought to the block input for the purpose of filtering input pulses E1 and E2 (marked in blue and green respectively) which have duration 1.25 µs. In order to test the functioning of filter for eliminating jitters, we placed three such impulses to the pulse E1. Filter's task was to eliminate them in order to prevent them from corrupting counter values.

Since proposed pulses E1 and E2 contain totally 13 regular rising and falling edges and block output has value of 13 (marked in red) in the end of the simulation, that suggests that jitters were eliminated, which proves correct functioning of all the elements used to make this design (filter, FSM and counter).

3.3. Velocity Estimator Block

The use of incremental encoders for measuring motor axis position θ does not allow direct measuring of motor axis rotation speed ω. Since rotation speed equals first derivative of the position value, procedure of acquiring speed ω is based on numerical differentiation of the position. Encoder signal accepting and processing block output is always available at each selection period. Namely at the time instant kT, the sample $\theta_k = \theta(kT)$ is available. Moreover, at the time instant $(k+1)T$, the sample $\theta_k + 1 = \theta(T(k+1))$ is also available, hence the information about the speed can be acquired through simple numerical differentiation:

$$\mathrm{Out}(z) = \mathrm{In}(z)\left(1 - z^{-1}\right). \tag{2}$$

Linearity of change in rotation speed is in most cases satisfactory because changes in load torque are usually values that change slowly in relation to general values of sampling time of speed regulator. Encoder signal processing block output is information about position that needs to be differentiated and it is brought to input In of differentiating block. Both of the described blocks, encoder interface and velocity estimation block, were experimentally tested by comparing output of installed tacho-generator (yellow colour) with output of velocity estimator (blue colour) as shown in Fig. 8. For this purposes tested object on its shaft had both encoder and tacho-generator installed.

It can be noted in Fig. 8 that the analog value corresponds to the velocity estimated via encoder measurement. Verification of design solutions was performed through experimental results. For this purpose, we used a motor with the maximum speed 1000 rpm. Both tachogenerator and incremental encoder with resolution 1000 ppr are connected to the motor axis. Both signals matching waveforms suggest that chosen concept for encoder signal processing for the purpose of acquiring information about position and velocity is correct [4].

3.4. Clock Generator and Sine Wave Generator Block

This block generates clock pulses (Fig. 9) needed for driving other blocks in developed toolbox. There is also a possibility to generate sine wave signal with various frequencies because these signals are used for testing and identification of object of control.

By using drop box menu a lot of fundamental sample times Ts (from 1.35 s to 10 µs) can be chosen. The obtained values of sample times Ts are the result of dividing basic clock frequency of FPGA circuit (50 MHz) with two. Obtained sample time values Ts cover wide range of applications of FPGA circuits in various areas in technology and industry. The encoder interface block can be driven with different clock frequencies in range from 650 µs to 0.16 µs depending on resolution of used encoder and rotational max shaft speed. The val-

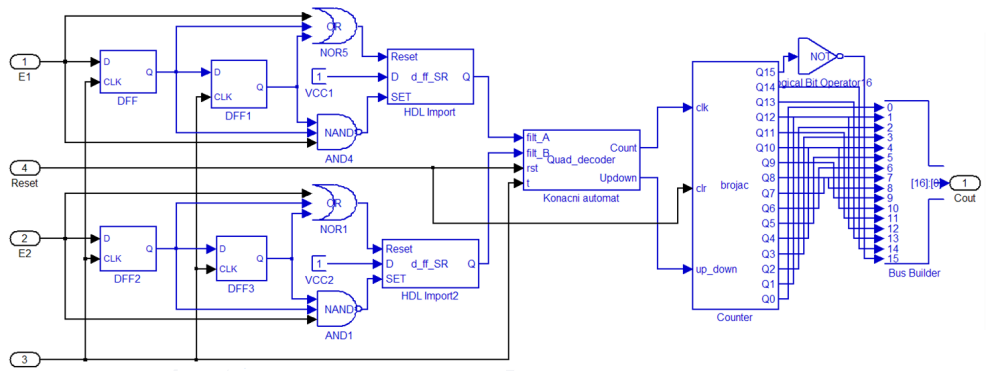

Fig. 6: Encoder signal accepting and processing design realized in Matlab/DSP Builder.

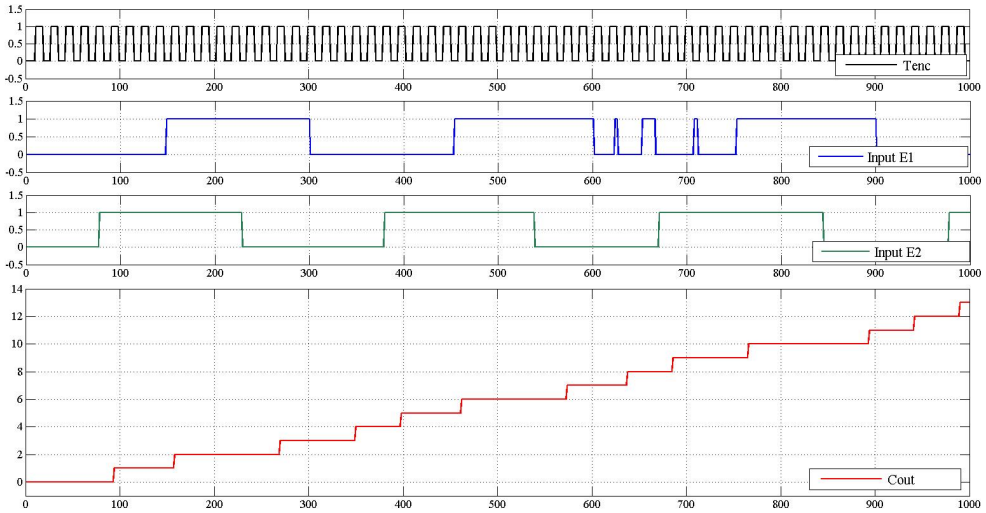

Fig. 7: Encoder signal accepting and processing design simulation result.

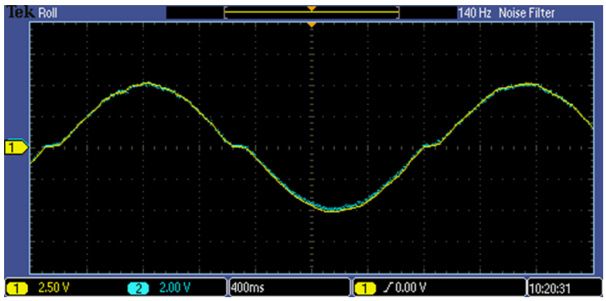

Fig. 8: Comparison between output of tacho-generator (yellow colour) with output of velocity estimator (blue colour) at sine velocity profile with magnitude of 1000 o/min.

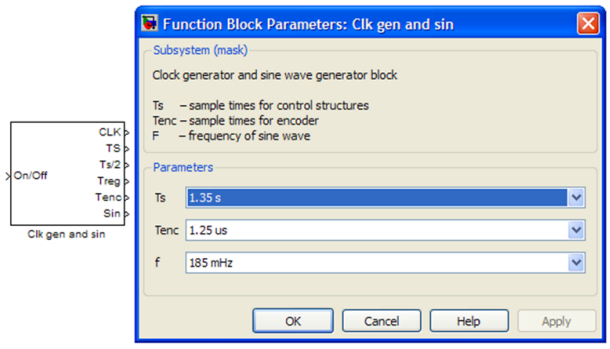

Fig. 9: Clock generator and sine wave generator block.

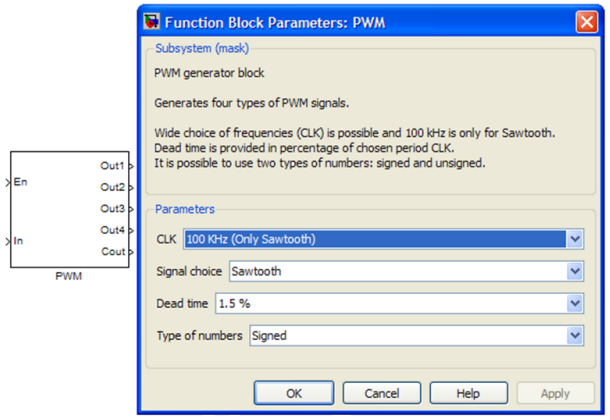

Fig. 10: PWM generator block.

ues of period sine signal that can be generated with this block are: 95 MHz, 185 MHz, 370 MHz and 740 MHz.

3.5. PWM Generator Block

In order to design control structures on FPGA circuits that control different types of power electronics converters it is mandatory to have PWM block that is shown in Fig. 10.

For different types of power electronics converters it was necessary to provide more types of PWM modulation patterns as well as different values of "dead time". It was defined as a percentage of frequency's reciprocal value, i.e. as a percentage of the period of PWM pulses. If it is necessary, it can be turned off if it is already implemented, for example, in the driver for controls switches in half-bridge and bridge topologies of power converters. Range of individual parameters in PWM block is shown in Tab. 1.

The input of PWM block *In* is accepting reference value which is compared with sawtooth/triangle carrier signal. On the right side of the block there are outputs marked from Out1 to Out4 which are used for obtaining PWM pulses. Figure 11 shows procedure of generating PWM signal by comparing triangle carrier

Tab. 1: Range of parameters in PWM block.

Frequency	Dead time	Modulation	Bus type
100 kHz	0	Sawtooth	Signed integer
50 kHz	0.75 %	Triangle	Unsigned integer
25 kHz	1.5 %	Bipolar	
12 kHz	2.3 %	Unipolar	
6 kHz	3.2 %		

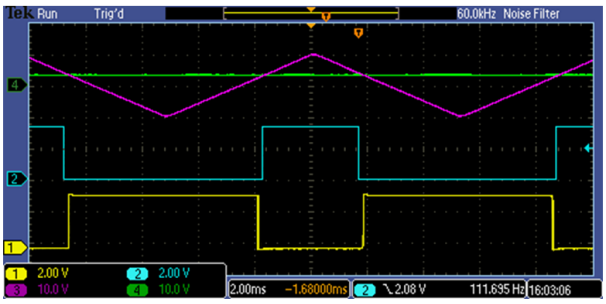

Fig. 11: Principle of generating PWM signal with triangle carrier signal.

signal with DC reference value and with inserted "dead time".

4. Simulation Results

The realized FPGA Real time toolbox with built blocks has been used for control structures for position and velocity design with DC motor as object of control. Figure 12 shows suggested typical control structure for position of one robot axis which is consisted of PID controller, discriminator of error and corresponding clock generator needed for operation of other blocks in control structure. The object of control has been modelled as moment of inertia J, where J is the sum of two moments of inertia, the first is DC motor drive and the second is connected load on shaft. Object of control parameters used in simulations is shown in Tab. 2.

There were two typical cases of application of control structures for position simulated in Matlab/DSP Builder: tracking reference sine trajectory (Fig. 13) and reaching constant reference values (Fig. 14). The abscissa axis presents the time specified by number of clock periods Ts.

Namely, when the simulation of digital circuits is performed, the fundamental clock time of FPGA circuit of 10 µs is presented on the abscissa axis, while the fundamental sample time Ts of simulated control structure of 1.25 µs is chosen.

This sample time is obtained by dividing basic clock frequency of 10 µs. Therefore, having in mind aforesaid, abscissa axis presents the time of 5 s because one second corresponds 10000 clock pulses of digital circuit. Suggested control structure for velocity was also simu-

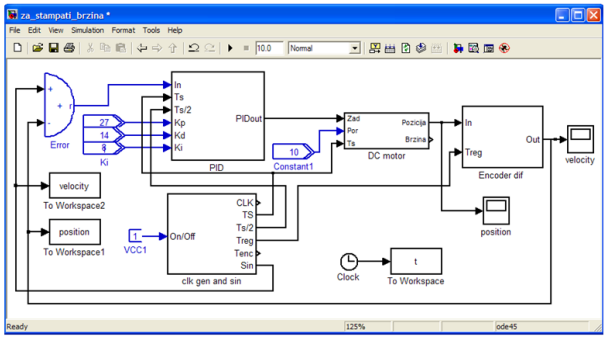

Fig. 12: Model of control structure for position/velocity realized in Matlab/DSP Builder.

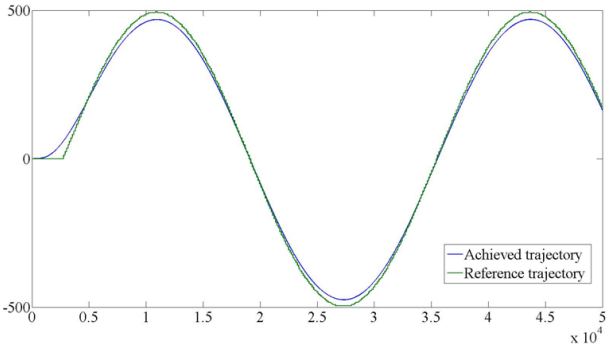

Fig. 13: Tracking reference sine trajectory.

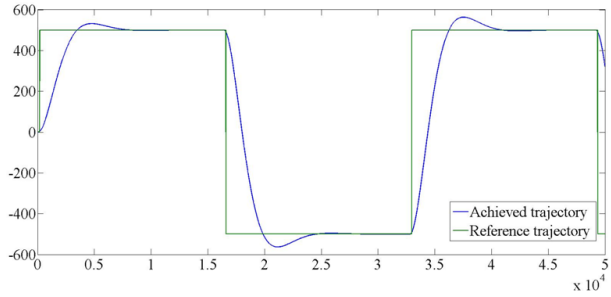

Fig. 14: Reaching constant reference value.

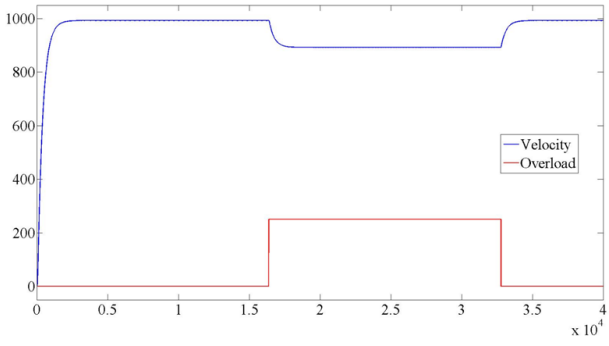

Fig. 15: Response of control structure for velocity obtained by simulation in Matlab/DSP Builder toolbox.

Tab. 2: Object of control parameters.

Parameters	Values
Resistance R_k	1.5 Ω
Inductance L_r	6 mH
C = k_m = k_c	0.167 Nm·A^{-1}
Moment of inertia J	0.002 kg·m^2

lated in Matlab and its response is shown in Fig. 15. It is evident from Fig. 15 that during overload velocity is somewhat reduced, which is not primary issue of this control structure and which is opposed to reduction of overshooting. From the Fig. 15 it is also clear that with appropriate choice of parameters fast response can be obtained in transient state without overshooting the velocity (blue colour). Upon termination of an overload (red colour) object of control reaches the reference speed also without overshooting the reference velocity.

5. Interface Boards

It was necessary to carry out functional verification of developed blocks in realized toolbox experimentally. For this purpose all mentioned simulated control structures were built on development board DE2 and tested on experimental apparatus (one axis of robot movement), [5], [8] and [9]. The problem that arises is how to connect a digital control structure with object of control. Since the FPGA is digital circuit that generates only digital signals, it is often necessary to convert control signal given in digital form to analog form (analog nature of actuators). Also, feedback signals from object of control can often be analog and there is need to convert these signals to digital form in order for them to be accepted on FPGA circuit. Object of control often supports TTL logic level for connection to other devices at its outputs, while FPGA circuits usually support LVTTL, so it is necessary to convert LVTTL to TTL and vice versa. For the purpose of connecting the DE2 development board with previously mentioned object of control and data presentation two interface boards were realized (Fig. 16). The basic functional requirements of the interface card are: DA conversion (3 convertors), AD conversion (2 convertors), TTL to LVTTL voltage conversion and vice versa. According to the manufacturer's specifications, signal on output of DA converter is available for 100 ns while the signal on output of AD converter is available for 100 µs. This means that the signal from the AD converter should be accepted within required 100 µs.

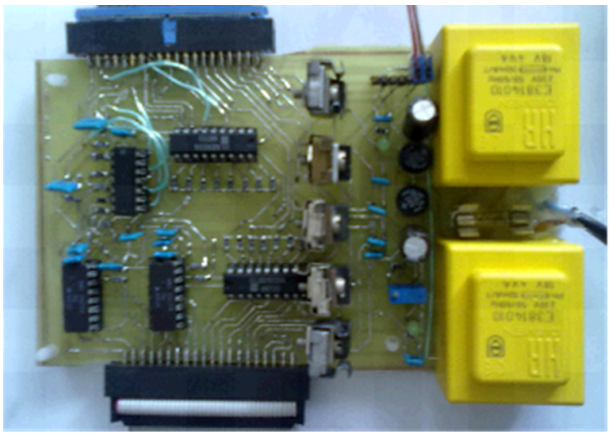

Fig. 16: Interface board.

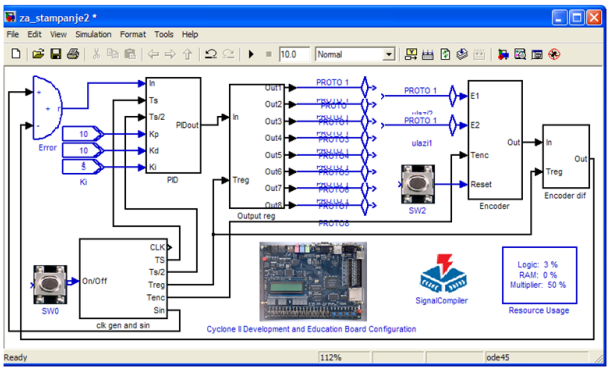

Fig. 17: Realized control structure for position/velocity in Matlab/DSP Builder environment.

6. Experimental Results

For the purpose of functional verification of developed blocks in realized toolbox and realized control structures for position and velocity it was necessary to carry it out experimentally. In Fig. 17 is shown control structure of one axis of robot movement which, with minor modifications, can be used for position or velocity control. If it is necessary to have PI control structure for velocity, then PID with parameter of differential part $K_d = 0$ should be used.

The following figures show the experimental responses of the realized control structures that have been previously developed during simulation phase. Figure 18 shows experimental responses of realized control structure for position in the task of tracking reference sine trajectory [5]. Signal coloured yellow presents reference sine trajectory while signal coloured blue presents the actual trajectory. The shown responses were obtained under the same conditions as in the simulations, i.e., the same nonlinear load directly coupled with motor shaft because used motor does not have gear box. Values on the ordinate axis represent the position of rotated shaft where the range of $\pm 90°$ corresponds to ± 500 pulse encoder.

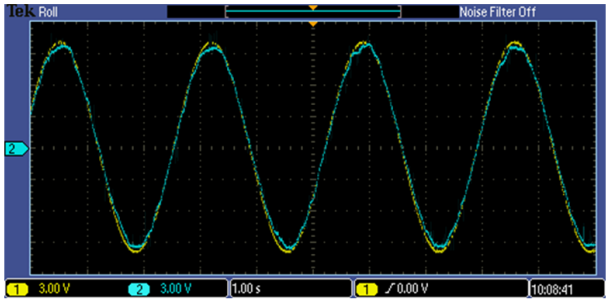

Fig. 18: Experimental response of control structure for position in the task of tracking reference sine trajectory 360 MHz ± 500 pulses.

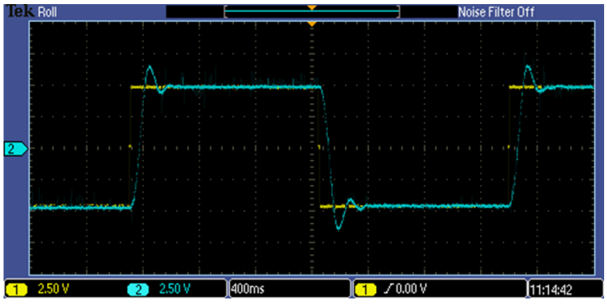

Fig. 19: Experimental response of control structure for position in the task of reaching reference value of position.

In Fig. 19 is given a response of the control structure for position in the task of reaching reference position value [5]. As it can be seen, in the stationary state there does not exist static error while in transient state there is some overshoot according to the applied criteria of selected parameters of PID controller. The reference value of position was changed with frequency of 360 MHz and number of encoder pulses ± 500 is corresponding shaft rotation of ± 450. In Fig. 19 is shown experimental response of control structure for in the task of reaching the shaft reference velocity. The figure shows three signals: velocity reference value of 1500 o/min (blue colour), achieved velocity (yellow color) and overload on the shaft (magenta colour). The occurence of overload on the shaft causes reducing shaft velocity, so the task of the control structure is to try to reach reference velocity regardless of the shaft load by increasing generated torque.

Upon removal of overload on the shaft, realized control structure for velocity provides the reaching of the reference velocity without overshooting as is shown in Fig. 20. Matching between experimental and simulated results confirms assumption that initial requirements for design blocks of FPGA Real Time toolbox were correct. Experimental response of control structure for velocity in the task of reaching the shaft reference velocity of 1500 o/min with short overload on shaft.

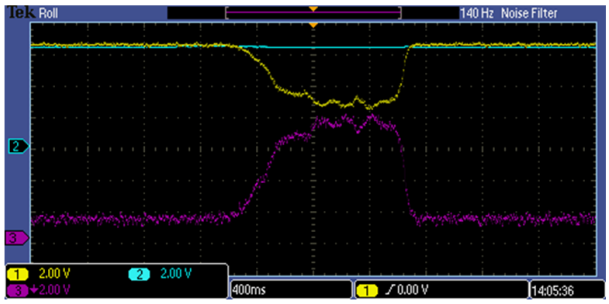

Fig. 20: Experimental response of control structure for position in the task of tracking reference sine trajectory 360 MHz ±500 pulses.

7. Conclusion

The main goal of the performed research described in this paper was the realization of an integrated environment, primarily toolbox in Matlab/DSP Builder, for development and testing of a wide class of control algorithms on FPGA platforms. The developed integral environment has software (Toolbox) and hardware (Interface card) component offering simple method for design of control algorithms without engaging in specificity of programming FPGA circuits. During the execution of research tasks basic objectives were obtained: FPGA Real Time was realized in Matlab/DSP Builder, functionality of toolbox building blocks was checked by simulations in Matlab, hardware components for the experimental verification of the built blocks were designed and implemented and finally built blocks were experimentally verified. The designed position/velocity control structures for position/velocity were tested on experimental platform (one axis robot) in the laboratory.

References

[1] BARLAS, T. and M. MOALLEM. INTECH. D*eveloping FPGA-based Embedded Controllers using Matlab/Simulink*. Simon Fraser University, 2009. Available at: `http://cdn.intechopen.com/pdfs-wm/10841.pdf`.

[2] JIANG, L. and W. WANG. The design of acquisition circuit for grating digital signal based on FPGA. In: *3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*. Chengdu: IEEE, 2010, pp. 134–137. ISBN 978-1-4244-6539-2. DOI: 10.1109/ICACTE.2010.5579399.

[3] JOKIC, D., S. LABURA and S. DORDEVIC. Projektovanje i realizacija kontrolera za robot puma 560. In: *Naucno-strucni Simpozijum INFOTEH*. Jahorina: University of East Sarajevo, 2010, pp. 105–109. ISBN 99938-624-2-8.

[4] JOKIC, D., S. LUBURA, S. LALE and D. LUKAC. Encoder signal processing on FPGA platform realized in Matlab/DSP Builder. In: *20th Telecommunications Forum (TELFOR)*. Belgrade: IEEE, 2012, pp. 1044–1047. ISBN 978-1-4673-2983-5. DOI: 10.1109/TELFOR.2012.6419389.

[5] JOKIC, D., S. LUBURA and M. SOJA. Closed control loop implementation for single robot axis on FPGA platform. In: *11th IFAC Conference on Programmable Devices and Embedded Systems*. Brno: IEEE, 2012, pp. 174–179. ISBN 978-3-902823-21-2. DOI: 10.3182/20120523-3-CZ-3015.00035.

[6] JORDAN, B. *Jordan Engineering Technologies: Electronics and embedded system arts* [online]. Available at: `http://www.jordandsp.com`.

[7] ATMEL CORPORATION. Atmel [online]. 2014. Available at: `http://www.atmel.com`.

[8] LUBURA, S., V. MITROVIC, M. SOJA and S. G. DORDEVIC. Projektovanje aparature za ispitivanje algoritama upravljanja distribuiranog pogona robota. In: *51. Conference ETRAN*. Herceg Novi: ETRAN, 2007, pp. 174–179. ISBN 978-8-680-50918-1.

[9] LUBURA, S., G. DORDEVIC and V. ZERBE. Aparatura za proucavanje procesa motornog ucenja u balistickim zadacima pogadanja mete. In: *VIII. International Symposium on Industrial Electronics*. Banja Luka: University of Banja Luka, 2010, pp. 279–283. ISBN 978-99955-46-03-8.

About Authors

Dejan JOKIC was born in Sarajevo, Bosnia and Herzegovina in 1980. He received his M.Sc. from University of East Sarajevo in 2012. His research interests include development of control algorithms of FPGA platforms and robotics.

Slobodan LABURA was born in Sarajevo, Bosnia and Herzegovina in 1969. He received M.Sc. and Ph.D. from University of East Sarajevo, Faculty of Electrical Engineering 2006 and 2009, respectively. His research interests include modeling and simulation of complex systems, development of control algorithms of DSP and FPGA platforms.

Stevan STANKOVSKI was born in Novi Sad,

Serbia 1962. He received M.Sc. and Ph.D. from University of Belgrade, Faculty of Electrical Engineering 1991 and 1994, respectively. His research interests include mechatronics, automation, control systems and artificial intelligence.