TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 19, No. 1, February 2021, pp. 220~228
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v19i1.16432

Embedded processor system for controllable period-width multichannel pulse width modulation signals

Mazin Rejab Khalil, Laith A. Mohammed
Engineering Technical College, Northern Technical University, Iraq

Article Info

Article history:

Received Apr 19, 2020
Revised Jul 27, 2020
Accepted Sep 24, 2020

Keywords:

Embedded design techniques
Processing system
Programmable logic
programming interfaces
Pulse width modulation

ABSTRACT

This paper proposes a sophisticated embedded processor system configured on zynq-xc7z020 field programmable gate array (FPGA) device for generating four channels pulse width modulation signals with variable duty cycles and periods using embedded design techniques. The main advantages of the technique are the high ability to perform a simultaneous control on period and pulse width of the generated signals and a high system design adaptation to choose the number of input/output channels. Controlling the the period and the pulse width is achieved by injecting a digital signal to the designed system to manipulate embedded timers' operation. Vivado design suite is used to develop the system hard ware in the integrated development environment where the processing unit and peripherals are instantiated and interconnected. A practical aplication program in C language is prepared to make the system act according to the target. The designed system can be used to drive multi-phase D.C to D.C convertors. The system performance is verified by using vivado logic analyzer and chipscope windows. The superiority of the proposed approach over other approaches is that it resulted in a multi-inputs/multi-outputs pulse width modulation system with high controllability on the pulse width and the period that ranges from 15 nsec to 60 sec.

Corresponding Author:

Mazin Rejab Khalil
Engineering Technical College
Northern Technical University
Maj moah. Mosul, Iraq
Email: mazinrk56@gmail.com

1. INTRODUCTION

Pulse width modulation (PWM) signals occupy a wide scope of applications in electrical drives and power electronics. Different algorithms and means were adopted for generating pwm signals according to the type of application and available resources. Embedded design techniques are a brilliant tool to design an embedded processor system, configure it on field programmable gate arrays (FPGAs) slice and program it to act according to the target that can be a pwm signal generator.

A PWM embedded module based on 8 bits 8051 microcontroller is proposed by [1], the PWM pulse is generated by programming the control register and duty cycle register. FPGAs based variable and multiple PWM signal is developed by [2] using comparators, counters and latching logics constructed by using verilog hardware description language (HDL). Using Nois II processor a new type of PWM peripheral was designed and accommodated by [3]. Trapezoidal modulated FPGAs based PWM controller is described by [4]. A design of microblaze processor system accommodated to generate single PWM signal using embedded design techniques was forwarded by [5]. A hard ware/software microblaze soft-core based system to generate PWM signal using clock frequency to control the pulse width is presented by [6]. Sinusoidal pulse width modulation system is constructed by [7] to act as AC voltage controller. An iterative reduction based heuristic algorithm with closed loop and space vector PWM control of a Z-source inverter implemented in hardware is discussed in [8]. A ratio based PWM technology in combination with proportional controller is adopted by [9] to achieve a smooth mid point potential controller. Generating PWM signal by comparing triangular wave with an adjustable direct current (DC) reference signal is performed by [10] to be used with D.C to D.C converter. Phase shift angle digital PWM (DPWM) for Phtovoltaic inverters is developed in [11], the proposed system is adapted to FPGAs platform. A PWM switching scheme for multiphase interleaved converter using field Programmable gate arrays (FPGAs) is proposed by [12], the duty cycle is adjusted by using assigned switches on Altera board. Space vector pulse width modulation (SVPWM) technique is described by [13], the basic principle of SVPWM development depends on synthesizing a rotating reference voltage vector. Different techniques were adopted in [14-17] to generate various types of PWM signals for different applications.

The main target of the above-mentioned works was to generate single or multi-channels PWM signals with constant period and variable pulse width. Alternatively, the goal of the work here is to generate single or multi-channels pwm signals with variable period and variable pulse width, the period and pulse width in each channel can be varied simultaneously in accordance with an external signal. This target is achieved by using embedded design techniques to construct a processor system that includes two Arm® Cortex™-A9 MP Core™ processors to be implemented on zync-xc7z020 FPGAs slice, the hardware is programmed by preparing C language software application that adapts the system to act in accordance with the target. The benefit of the suggested approach is that it results in a simple, applicable and reconfigurable system. The designed system can also be accommodated to generate signals required by the systems discussed in [18-20]. The remaining of the paper is organized as follows. In section 2, the research method is explained to show the system hardware construction that includes intellectual property (IP) cores instantiation, IP cores interconnection and FPGA slice configuration. In the second part of section 2, the paper deals with system software development. In section 3, the results are displayed and discussed. The conclusions and future works are given in section 4.

2. RESEARCH METHOD

The system design implies three stages. In the first stage the hardware part is constructed, while the second stage includes programming the hardware to operate according to the target and the third stage is specified for debugging the system performance. The zynq FPGA slice consists of processing system (PS) with two ARM Cortex™-A9 processor cores, programmable logic (PL) and a number of intellectual properties components (IPs) [21, 22]. The system hardware design depends on instantiating the intellectual properties (IP) cores to PL and attaching it to PS to constitute the desired processor system. On chip debugging schemes were used to display the generated signals [23-25].

2.1. System hardware construction

The hardware part of the system is developed by using the IP integrator tool available in the vivado integrated development environment (IDE) [21, 22]. Figure 1 shows the block diagram of the zynq FPGA device [26]. The hardware construction procedure implies three steps, IP cores instantiation, IP core interconnection and FPGAs slice configuration.

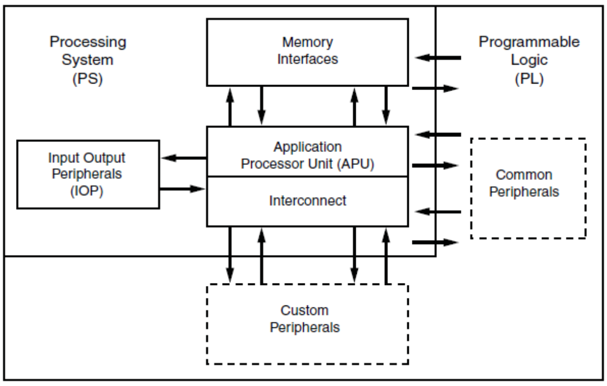


Figure 1. The block diagram of of zynq FPGA device

2.1.1. IP cores instantiation

In the first step a processing unit, block RAM (BRAM) controller IP core, General purpose input/output (GPIO) IP core, and four timers IP cores (timer_0, timer_1, timer_2 and timer_3) are instantiated to the IP integrator diagram as shown in Figure 2. The processing unit in PS domain contains two processor cores, each of them includes 32KB instruction cache, 32 KB data cache, floating point unit and memory management unit (MMU) to provide access to the one mega byte dual data rate dynamic RAM (DDR3) available on the evaluation board [21, 26-30]. The block RAM is 64 KB two ports memory, it is mostly used as a boot memory. The GPIO unit is a custom input/output unit that deals with data width ranging between (1-64) bits, the on-line data acquired by this unit is used to determine the duty cycle and pulse width of each output control signal [31, 32]. The timers are used to generate the output control pulses, each timer module consists of two 32 bits counters (counter 0 and counter 1) as shown in Figure 3, each counter is associated with a load register (LR0 and LR1 for counter 0 and counter 1 respectively) that is used to hold the count value of the counter and a control/status register (CR0 and CR1 for counter 0 and counter 1 respectively) to control the operation mode of the counter [33]. Each counter generates a single pulse at the expiration of count interval, the two counters are used to generate the output signal in the form of pulse width modulation (pwm) where counter 0 sets the period and counter 1 specifies the high time of the output signal [31, 33]. The timer's operation is controlled by programming the control/status registers by the prepared software program. The data acquired by the GPIO is to be transferred to the load registers of the counters via the prepared program to set the output signal specifications.

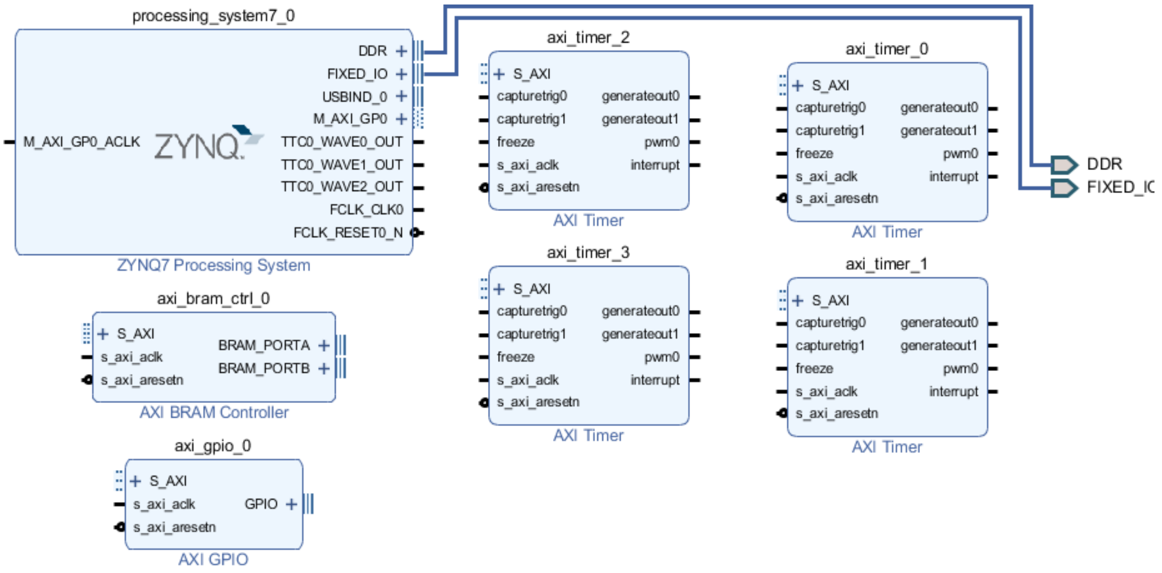


Figure 2. Hardware components of the designed processor system

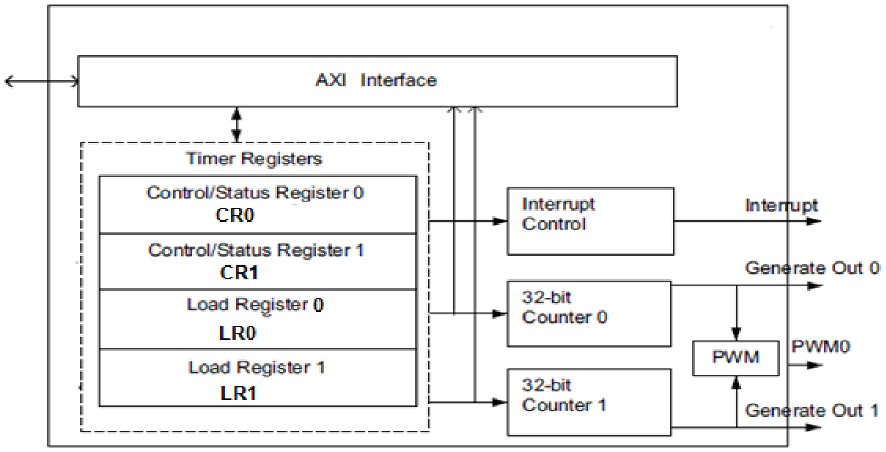


Figure 3. The timer module

2.1.2. IP cores interconnection

In the second step the connection operation is performed in the IP integrator environment [21, 22]. Figure 4 shows the resultant system hardware, the processing system is connected to the peripherals by AXI-interconnect unit [34, 35]. The resultant system contains four output channels (pwm0, pwmo_c1, pwm0_c2, pwm0_c3), each channel is used to drive a DC to DC convertor with suitable control signal. Figure 5 displays the address map of the designed processor system.

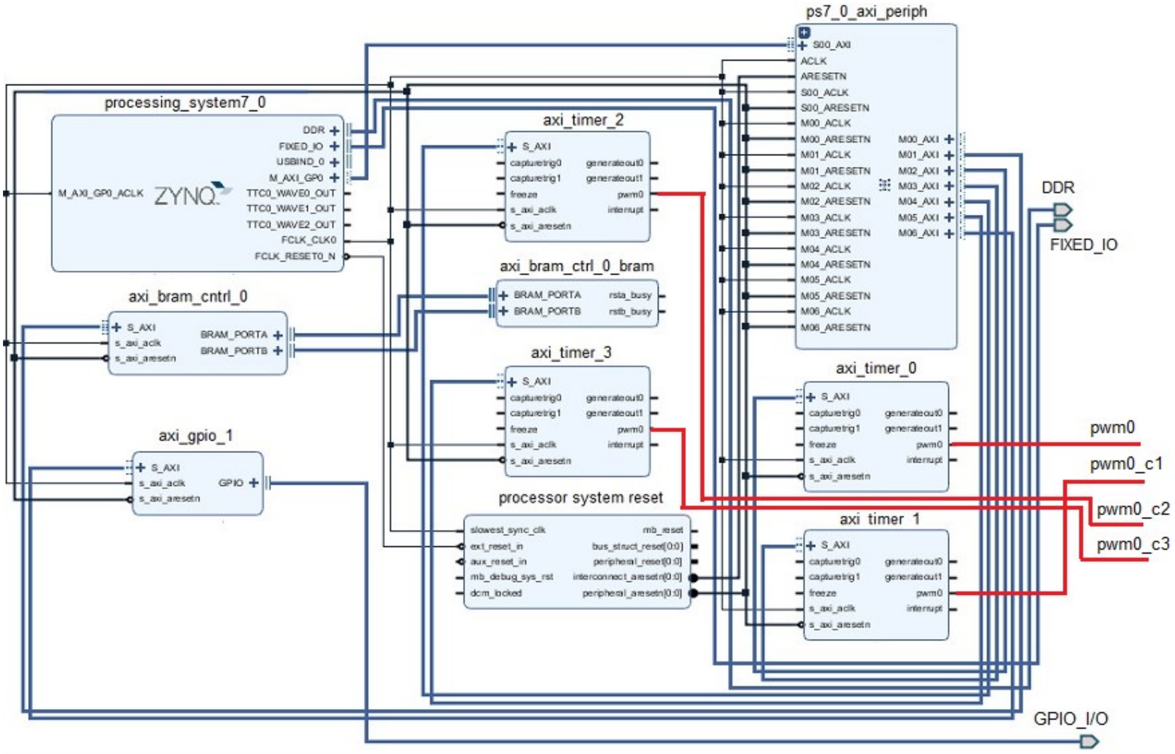
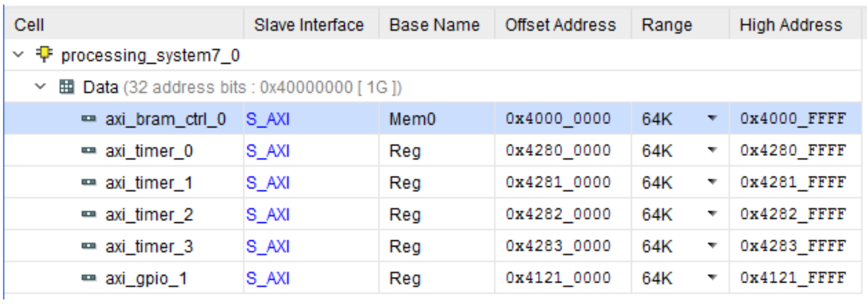


Figure 4. The designed embedded processor system hardware



Cell	Slave Interface	Base Name	Offset Address	Range	High Address
processing_system7_0					
Data (32 address bits : 0x40000000 [1G])					
axi_bram_ctrl_0	S_AXI	Mem0	0x4000_0000	64K	0x4000_FFFF
axi_timer_0	S_AXI	Reg	0x4280_0000	64K	0x4280_FFFF
axi_timer_1	S_AXI	Reg	0x4281_0000	64K	0x4281_FFFF
axi_timer_2	S_AXI	Reg	0x4282_0000	64K	0x4282_FFFF
axi_timer_3	S_AXI	Reg	0x4283_0000	64K	0x4283_FFFF
axi_gpio_1	S_AXI	Reg	0x4121_0000	64K	0x4121_FFFF

Figure 5. The address map of the designed processor system

2.1.3. FPGAs slice configuration

The third step implies configuring the FPGAs slice with system bit stream through synthesis, implementation, bit stream generation and down loading operations [21, 22]. Xilinx synthesis technology (XST) tool is used in synthesis stage to generate an industry-standard electronic data interchange format (EDIF) file. The implementation stage implies converting the generated (EDIF) file into a physical file format. The physical information contained in native circuit description (NCD) file is used to create a bit stream file that is necessary to program the FPGAs slice.

2.2. System software development

The software part of the system is developed using the software development kit (SDK) tools available in vivado IDE, the processor system is programmed to generate the required control signals using C language [36]. The flow chart of the prepared program part that is responsible for generating the PWM signals is shown in Figure 6. The load registers LR0 and LR1 of each timer are loaded with arrays A0 and A1 that holds the period and high time values respectively in a fashion shown in Figure 6. The data (A0 and A1 arrays) acquisition is performed via the input/output port GPIO_IO using certain application programming interfaces (APIs) prepared for this purpose with C language [36]. The GPIO_IO port is composed of two channels, channel 1 is specified for acquiring data of A0 array and channel 2 is specified to acquire data of A1 array in the fashion shown in Table 1. Figure 7 shows the control/status register diagram for each counter while Table 2 explains the used bits definition of the control/status register [33]. The control/status register is set as shown in Figure 7 to generate PWM signal according to the register value (register value=0x214).

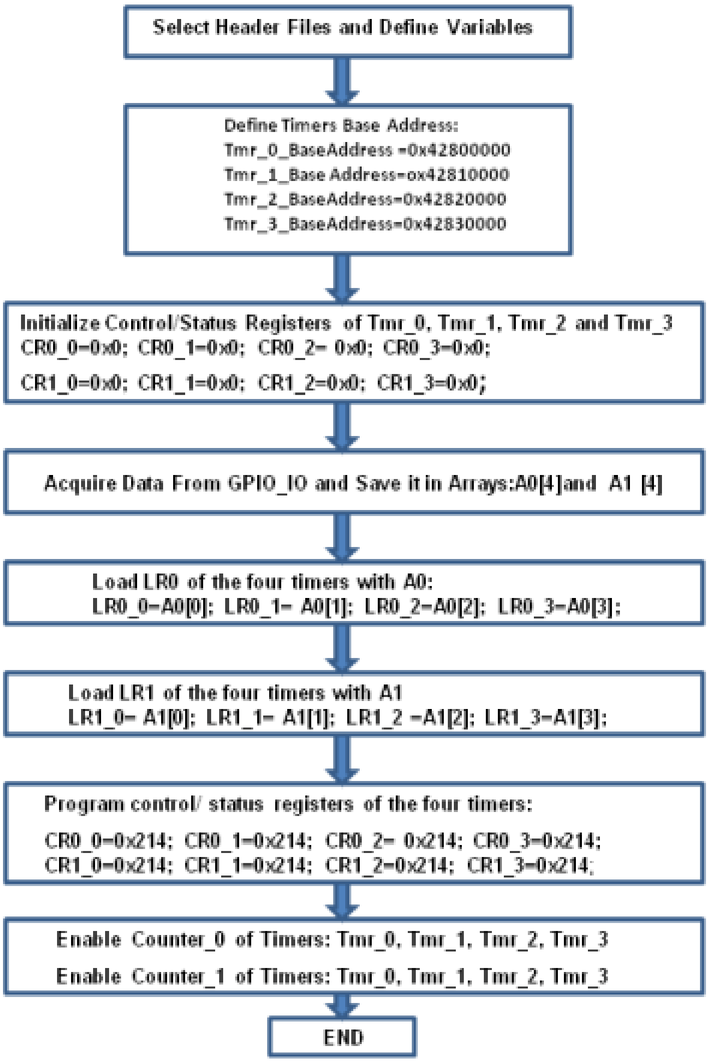

Select Header Files and Define Variables
Define Timers Base Address:
Tmr_0_BaseAddress =0x42800000
Tmr_1_Base Address=ox42810000
Tmr_2_BaseAddress=0x42820000
Tmr_3_BaseAddress=0x42830000
Initialize Control/Status Registers of Tmr_0, Tmr_1, Tmr_2 and Tmr_3
CR0_0=0x0; CR0_1=0x0; CR0_2= 0x0; CR0_3=0x0;
CR1_0=0x0; CR1_1=0x0; CR1_2=0x0; CR1_3=0x0;
Acquire Data From GPIO_IO and Save it in Arrays:A0[4]and A1 [4]
Load LR0 of the four timers with A0:
LR0_0=A0[0]; LR0_1= A0[1]; LR0_2=A0[2]; LR0_3=A0[3];
Load LR1 of the four timers with A1
LR1_0= A1[0]; LR1_1= A1[1]; LR1_2 =A1[2]; LR1_3=A1[3];
Program control/ status registers of the four timers:
CR0_0=0x214; CR0_1=0x214; CR0_2= 0x214; CR0_3=0x214;
CR1_0=0x214; CR1_1=0x214; CR1_2=0x214; CR1_3=0x214;
Enable Counter_0 of Timers: Tmr_0, Tmr_1, Tmr_2, Tmr_3
Enable Counter_1 of Timers: Tmr_0, Tmr_1, Tmr_2, Tmr_3
END

Figure 6. The flowchart of the prepared application for generating PWM signals

Table 1. Data acquisition mode

Read Cycle	Acquired Data from Channel 1	Acquired Data from Channel 2
1	A0[0]→LR0 of timer_0	A1[0]→LR1 of timer_0
2	A0[1]→LR0 of timer_1	A1[1]→LR1 of timer_1
3	A0[2]→LR0 of timer_2	A1[2]→LR1 of timer_2
4	A0[3]→LR0 of timer_3	A1[3]→LR1 of timer_3

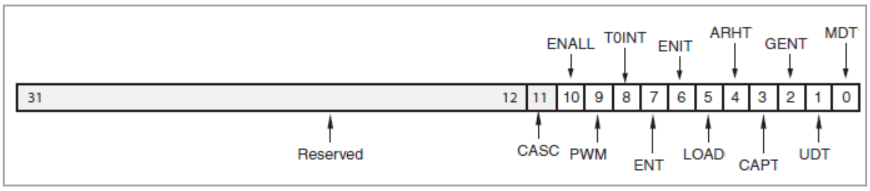


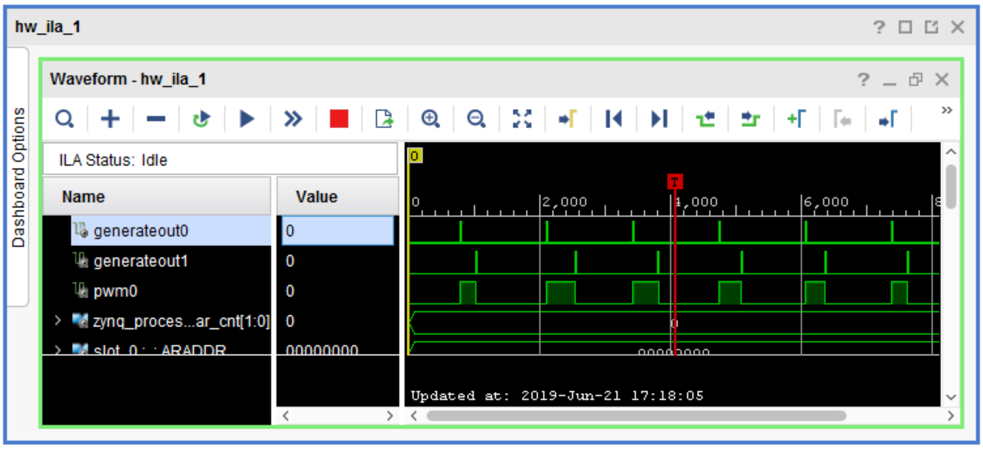
Figure 7. Control/status register diagram

Table 2. Control/status register bit definition

Bit	Name	Description
9	Pwm	0= Disable pwm, 1=Enable pwm
7	ENT	0= Disable timer, 1=Enable timer
6	ENIT	0= Disable Interrupt, 1= Enable nterrupt
5	Load	0= No load, 1= load timer with value in LR
4	ARHT	Auto Reload/Hold, 0= Hold counter, 1=Reload counter with value in LR
2	GENT	0= Disable external generate signal, 1= Enable external generate signal
1	UDT	0= Up count, 1 = Down count
0	MDT	Timer Mode, 0= Generate, 1= Capture

3. RESULTS AND ANALYSIS

Figure 8 displays a pulse width modulation signal (pwm0) generated by timer_0 on hardware manager integrated logic analyzer (hw_ila) window, the generateout0 and generateout1 signals are one clock pulses generated by counter 0 and counter 1 respectively. It is seen that the signal generateout0 sets the period while the on time of the pwm0 pulse is specified by both generateout0 and generateout1signals. At the expiration of the count interval of counter 0 a single pulse of generateout0 is released to initiate the on time of pwm0 and at the expiration of count interval of counter 1 a single pulse of generateout1 is released to terminate the on time of pwm0. The count interval of each counter depends on the number stored in LR0 and LR1 registers that contain A0[0] and A1[0] respectively. As A0[0] and A1[0] are acquired by the input/output port GPIO_IO, the data fed to the designed system via this port represents a control signals that control the period and pulse width of the generated signal (pwm0). Figure 8 shows that the control signal can vary the width of consecutive pulses resulting in a high controllability pwm generating system. The pwm signals generated by the four timers can be controlled in the same manner as pwm0.



ILA Status: Idle

Name	Value
generateout0	0
generateout1	0
pwm0	0
zynq_proces...ar_cnt[1:0]	0
slot_0 : ARADDR	00000000

Updated at: 2019-Jun-21 17:18:05

Figure 8. Pulse width modulation signal generated by timer_0

Figure 9 depicts four pwm signals (pwm0, pwm0_c1, pwm0_c2 and pwm0_c3) generated by the system four timers with different pulse width displayed on ILA chipscope window. The period and high time of each generated pwm signal is specified by the values injected in LR0 and LR1 registers of each timer in a style shown in Table 1. Figure 10 shows a new set of pwm pulses generated by the designed system after introducing new data set of A0 and A1 at sample 3225 resulting in a four channels pwm system with high flexibility to vary the period and pulse width of any pulse or set of pulses according to the data (external control signal) acquired by the input/output port GPIO_IO.

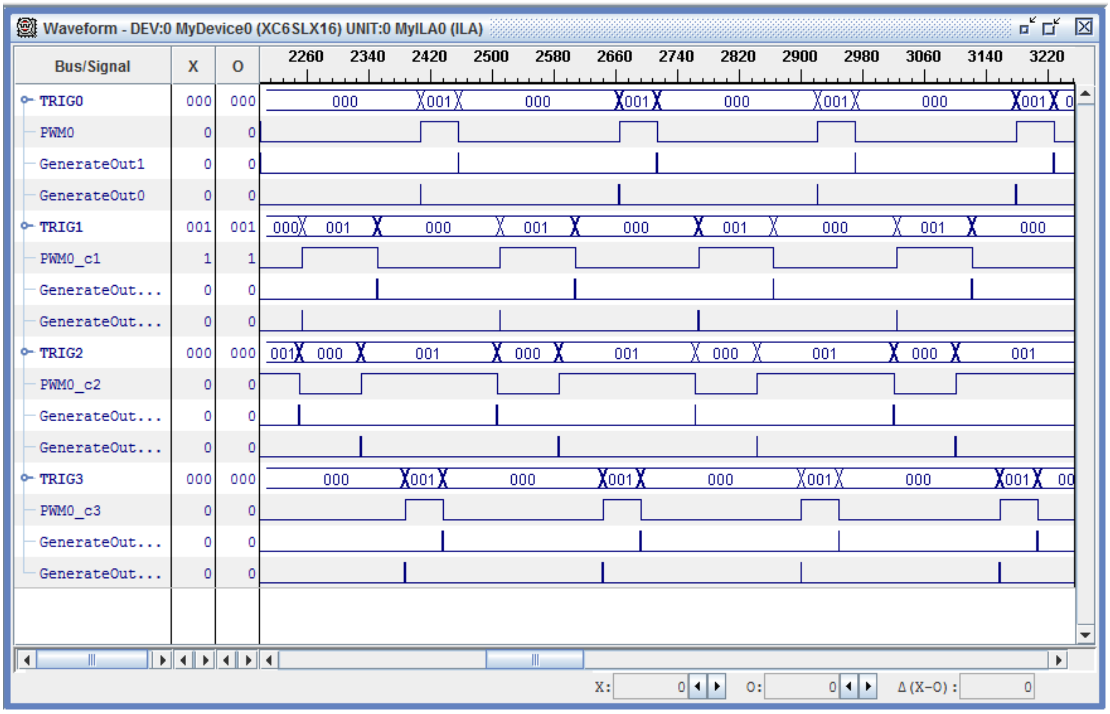


Figure 9. pwm pulses generated by the by the system four timers (pwm0-pwm0_c3)

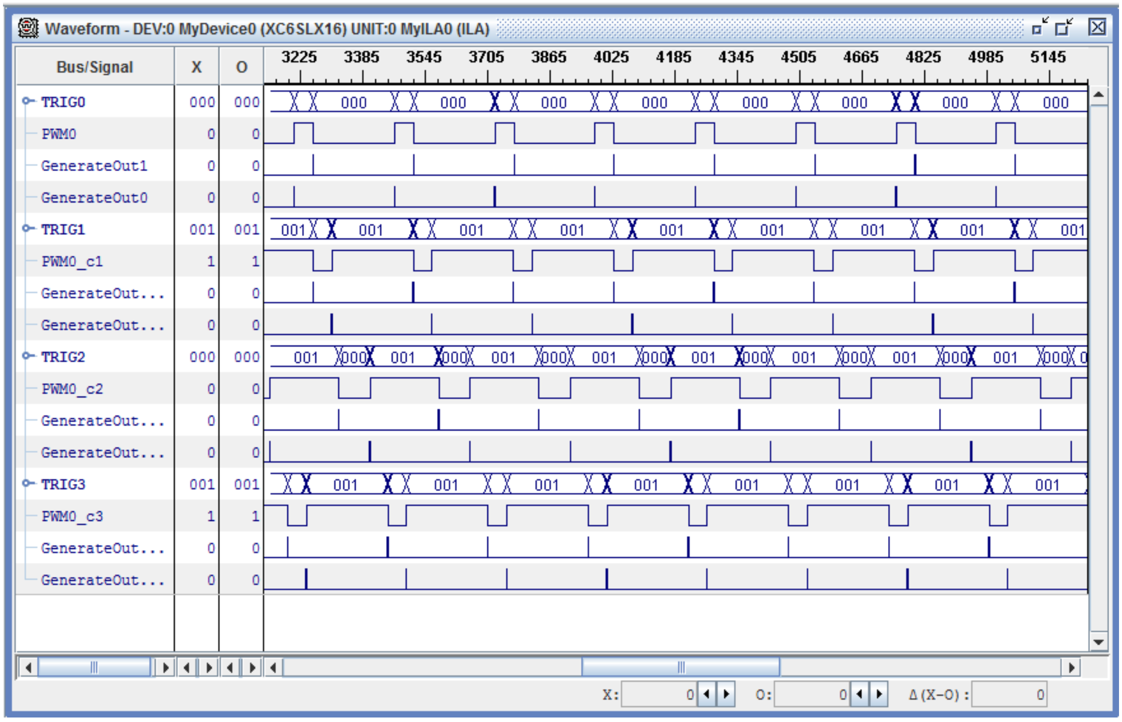


Figure 10. pwm pulses generated after introducing new set of control signals

Since the used counters are 32 bits counters, the achievable range of period can be between 15 nsec to 60 seconds. Table 3 shows the high time and period for five consecutive pwm signals that are achieved by the designed system at pwm0 channel, the periods are chosen deliberately such that their sum is 20 msec. The obtained results indicate that the designed system implies high ability to realize a simultaneous control on the period and duty factor of four pwm channels pulse by pulse in response to external effect. This is a significant improvement over other to date pwm systems that offer a control on period or duty factor.

Table 3. Obtained high time and period for five consecutive pulses

Pulse Number	High Time(μsec)	Period (μsec)
1	2860	5146
2	162	4278
3	576	3436
4	2286	2448
5	4116	4692

4. CONCLUSIONS

The main goal of the current paper was to develop a new sophisticated method for designing and implementing a processor system that can be configured on zynq-xc7z020 FPGA device to generate four channels pwm signals with variable period and pulse width. According to the obtained results, it can be concluded that; four pwm signals can be generated simultaneously with zynq-xc7z020 FPGAs slice using embedded design techniques. As the number of generated signals depends mainly on the available resources of the FPGAs slice, the number of the generated pwm signal trains can be extended accordingly. The period and pulse width of each pwm signal can be varied in conformity with external signal that can be acquired through an input/output port yielding a controllable pwm signal generating system with (15 nsec-60 sec) achievable range of period. The pulse width and period can be varied pulse by pulse or group of pulses by group for each train simultaneously resulting in a system with high controllability.

The results obtained from this paper enhances the ability to create a pwm signal controller whose response is accommodated with an external effect to control converter circuits operation. The novelty of the work is characterized by the designed system ability to perform a simultaneous variation in pulse width and period, pulse by pulse. Further experimental investigations are suggested to generate 8-16 channels pwm system using the same techniques adopted in this work.

REFERENCES

[1] Y. Li Hu and W. Wang, "Design of PWM Controller in a MCS-51 Compatible MCU," *International Symposium on High Density Packaging and Micro system Integration,* pp. 1-3, 2007.

[2] S. Pong, *et al.*, "Digital Technique to Generate Variable and Multiple PWM Signals," *Second International Conference on Innovative Computing, Information and Control (ICICIC 2007)*, pp. 457-463, 2007.

[3] Y. Xu and M. Xiang, "Design A New Type PWM Peripherals in Nios II," *WRI World Congress on Computer Science and Information Engineering*, vol. 1-7, pp. 442-446, 2009.

[4] A. Agarwal, *et al.*, "Design of FPGA Based Controller for Trapezoidal Modulated Cycloinverter," *Fifth International Conference Information and Automation for Sustainability,* pp. 479-483, 2010.

[5] M. Khalil and S. Mahmood, "Designing of a Pulse Width Modulation System Using Embedded System Design Techniques," *Journal of Theoretical and Applied Information Technology,* vol.49, pp. 101-106, 2013.

[6] A. Zemmouri, *et al.*, " Design and Implementation of Pulse Width Modulation Using Hardware/Software Microblaze Soft-Core," *International Journal of Power Electronics and Drive System (IJPEDS),* vol. 8, no. 1, pp. 167-175, 2017.

[7] N. Murali and V. Balaji, "Real Time FPGA Implementation of PWM Chopper Fed Capacitor Run Induction Motors," *International Journal of Reconfiguable and Embedded Systems,* vol. 7, no. 3, pp. 138-148, 2018.

[8] A. Yilmaz and B. Erkmen, "FPGA-Based Space Vector PWM and Closed Loop Controllers Design for the Z Source Inverter," *IEEE Access,* vol. 7, pp. 130865-130873, 2019.

[9] F. Guangjie and L. Xiingpeng, "Research of a Neutral- Point Potential Control of a Three- level Inverter," *Indonesian Journal of Electirical Engineering and Computer Science,* vol. 14, no. 1, pp. 20-28, 2019.

[10] R. Baharom and A. Awang, "Development of Four Quadrant Operation of D.C to D.C Converter Using Single Phase Matrix Converter," *Indonesian Journal of Electirical Engineering and Computer Science ,* vol. 6, no. 3, pp. 1249-1256, 2019.

[11] L. Hassaine and M. Bengourina, "Design and Digital implementation of Power Control Strategy for Grid Connected Phtovoltaic Inverter," *International Journal of Power Electronics and Drive Systems,* vol. 10, no. 3, pp. 1564-1574, 2019.

[12] A. Gani, *et al.*, "Design and development of PWM Switching for 5-level multiphase Interleaved DC/DC Boost Converter," *Indonesian Journal of Electirical Engineering and Computer Science,* vol. 17, no. 1, pp. 131-140, 2020.

[13] Kadum, "PWM Control Techniques for Three Phase Three level Inverter drives," *TELKOMNIKA Telecommunication Computing Electronics and Control,* vol. 18, no. 1, pp. 519-521, 2020.

[14] Bharatiraja, *et al.,*"Investigation of Slim Type BLDC Motor Drive With Torque Ripple Minimization Using Abridged Space- Vector PWM Control Method," *International Journal of Power Electronics and Drive Systems,* vol. 8, pp. 593-600, 2017.

[15] Cuppuswamy and T. Raghavendiran, "Developed Z-Source H-Bridge Multilevel Inveter with Reduced Components for Speed Control of Induction Motor," *International Journal of Robotics and Automation,* vol. 7, no. 3, pp. 205-214, 2018.

[16] R. Palanisamy and K. Vejayakumar, "Switching Pulse Generation for DC-DC Boost Converter Using Xilinx- ISE With FPGA Processor," *International Journal of Reconfigurable and Embedded Systems,* vol. 8, pp. 81-85, 2019.

[17] Perez, *et al.*, "On the Dynamic Behavior of The Current in The Condenser of a Boost Converter Controlled With ZAD," *TELKOMNIKA Telecommunication Computing Electronics and Control,* vol. 18, no. 3, pp. 1678-1687, 2020.

[18] Sunita, *et al.*, "FPGA Based Photovoltaic (PV) Inverter with SPWM Algorithm for Photovoltaic System," *5th International Conference on Wireless Networks and Embedded Systems (WECON)*, pp. 1-5, 2017.

[19] H. Hatas, *et al.*, "FPGA Implementation of SPWM for Cascaded Multilevel Inverter by Using XSG," *4th International Conference on Power Electronics and their Applications (ICPEA),* pp. 1-6, 2019.

[20] Y. Li, *et al.*, "A modular Design Approach to Provide Exhaustive Carrier- Based PWM Patterns for Multilevel ANPC Converters," *IEEE Transactions on Industry Application,* vol. 55, no. 5, pp. 5032-5044, 2019.

[21] Xilinx, "Vivado Design Suite Tutorial: Embedded Processor Hardware Design (UG940)," 2018. [Online]. Available: https://www.xilinx.com/cgi-bin/docs/rdoc?v=2018.2;d=ug940-vivado-tutorial-embedded-design.pdf

[22] O. Baans and A. Jambek, "Implementation of an Arm - Based System Using a Xilinx Zynq Soc," *Indonesian Journal of Electirical Engineering and Computer Science,* vol. 13, no. 2, pp. 485-491, 2019.

[23] P. Acharya and M. Rani, "Berger Code Based Concurrent Online Self- Testing of Embedded Processor," *International Journal of Reconfigurable and Embedded Systems ,* vol. 39, no. 11, pp. 1-6, 2018.

[24] M. Hossain and W. Lee, "Integration Testing Based on Indirect Interaction for Embedded System," *International Journal of Reconfigurable and Embedded Systems,* vol. 8, no. 2, pp. 86-98, 2019.

[25] Suryawan, B. Handaga and A. Basith, "On- Chip Debugging for Microprocessor Design," *TELKOMNIKA Telecommunication Computing Electronics and Control,* vol. 18, no. 3, pp. 1229-1236, 2020.

[26] Xilinx, "Zc702Evaluation Board for the Zync-7000, XC7z020SOC User Guide (UG850)," 2018. [Online]. Available: https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug850-zync-7000 evaluation board.pdf

[27] Xilinx, "Local Memory Bus BRAM Controller, Data Sheet (DS452*),"* 2010. [Online]. Available: http://www.xilinx.com/support/documentation/1p documentation/lmb bram if cntlr.pdf

[28] M. Khalil and M. Hamdoon, "Development of a Scheme to Connect PC Monitor to Soft Core Processor," *The Mediterranean Journal of Computers and Networks*, vol. 7, pp. 178-184, 2011.

[29] S. Bahandary and S. Pujari, "HW-SW Co-Design On Zynq SoC Case Study:Simple Miniature DC Motor Speed Control System," *International Journal of Applied Engineering Research,* vol. 13, no. 12, pp. 10741-10748, 2018.

[30] H. Pandia, M. Rangapariya and J. Rajput, "Implement Embedded Controller Using FPGA," *International Journal of Reconfigurable and Embedded Systems,* vol. 8, no. 2, pp. 130-144, 2019.

[31] M. Khalil, "Design of an Interruptible Soft Core Processor System to Compute Discrete Cosine Transform of Audio Signals," *International Conference on Electrical, Communication, Computer, Power and Control Engineering (ICECCPCE -13). First Annual IEEE*, pp. 1-6, 2013.

[32] Xilinx, "LogiCORE™ AXI GPIO Product Guide (PG144)," 2016. [Online]. Available: https://www.xilinx.com/cgi-bin/docs/ipdoc?c=axi_gpio;v=latest;d=pg079-axi-gpio.pdf

[33] Xilinx, "LogiCORE™ AXI Timer Product Guide (PG079)," 2016. [Online]. Available: https://www.xilinx.com/cgi-bin/docs/ipdoc?c=axi_timer;v=latest;d=pg079-axi-timer.pdf

[34] M. Khalil and R. Mahmood, **"**Designing Embedded AXI Based Direct Memory Access System," *International Journal of Enhanced Research in Science, Technology & Engineering***,** vol. 4, no. 8, pp. 123-129, 2015.

[35] Xilinx, "LogiCORE™ IP AXI Interconnect Product Guide (PG059),"2018. [Online]. Available: https://www.xilinx.com/cgi-bin/docs/ipdoc?c=axi_interconnect;v=latest;d=pg059-axi-interconnect.pdf

[36] Xilinx, "Xilinx Software Development Kit (SDK) User Guide: System Performance Analysis (UG1145)," 2018. [Online]. Available: https://www.xilinx.com/cgi-bin/docs/rdoc?v=2018.2;d=ug1145-sdk-system-performance.pdf