Efficient FPGA Implementation of Field Oriented Control for 3-Phase Machine Drives

Burak Tufekci[1,3], Bugra Onal[2,3], Hamza Dere[2,3], and H. Fatih Ugurdag[2]

[1]CS Dept., Ozyegin University
[2]EE Dept., Ozyegin University
[3]Semimobility Teknoloji Ltd.
Istanbul, Turkey
burak.tufekci.17309@ozu.edu.tr

***Abstract*—This paper presents an FPGA implementation of Field Oriented Control (FOC) method with high switching frequency for 3-phase machine drives. A common architecture has been constructed for both BrushLess DC motors (BLDC) and Permanent Magnet Synchronous Motors (PMSM). For this purpose, the controller module has been implemented by using a hardware efficient algorithm, namely, Coordinate Rotation Digital Computer (CORDIC). The result of this implementation has been compared with the literature, and we claim that this paper's FPGA design has better performance in terms of area and speed with respect to other FPGA-based FOC designs.**

***Index Terms*—Motor Control, FPGA, FOC, BLDC, PMSM, CORDIC**

I. Introduction

Developments in semiconductor technology lead to energy-efficient and high frequencies power switches [1]. Such devices enabled us to come up with high frequency power system design solutions. High-frequency approach has many benefits in 3-phase (3-ϕ) motor drive applications [2]. Among these benefits can be better motor efficiency, low-cost filtering, lower torque ripple and faster control response. While higher frequencies of Pulse Width Modulation (PWM) have these advantages, they also cause voltage reflection and motor insulation breakdown issues at the motor terminals. Therefore, the operating PWM frequency and the type of the motor must be examined carefully.

Increasing the PWM frequency may not be easy for any setup. While the operating frequencies of microcontrollers are sufficient to exceed our device frequencies, it does not change the fact that microcontrollers can sometimes reboot and sequential iteration process can often take too long to measure non-linear operations. On the other hand, algorithm development on microcontrollers is faster than other semiconductor platforms.

FPGAs are superior to microcontrollers in many areas in terms of latency, connectivity, and energy consumption. Latencies of FPGA implementations can be 1 millisecond or even less, while even with the best CPUs introduce latency of approximately 50 milliseconds. Furthermore, because FPGAs do not contain any cache or OS, the delays are deterministic. Because input and output can be directly connected to FPGAs, this can enable high bandwidth implementations. FPGAs pin voltages are usually adjustable, they are very good at minimizing energy consumption.

Since both the speed control range of BrushLess DC motor (BLDC) and Permanent Magnet Synchronous Motor (PMSM) is large and the energy losses are lower in high-frequency applications compared to other motor types, PMSM and BLDC provide an ideal environment for the testing purposes of this work. Besides, FPGAs in a high speed design can respond better and FPGAs control mechanism is safer, they may be a good alternative for testing.

In the literature, Kung et al. proposed an FPGA-based approach to speed control with FOC [3]. An Sliding Mode Observer (SMO) design has been implemented in their work using a sensorless FOC and a phase-locked loop. The speed information has been generated by the user using the NIOS II processor and all other topologies have been implemented in the FPGA. According to the results obtained by the authors, the back-emf graphics in the transition from stop to acceleration can be smoothed by their own approach. Suneeta et al. have introduced FPGA-based control of 3-ϕ BLDC [4]. It has been shown to be more powerful and safer than microcontroller-based electric motor control because of the high design freedoms offered by FPGA-based electric motor controls. Otherwise, because of faster design development, microcontroller-based control is more powerful than FPGA-based controller, and is also cheaper than FPGA-based controller. Hence the choice of controller based on FPGA or microcontroller based control depends on system requirements.

Babu and Athul have used the PI-controller viewpoint to execute FOC on asynchronous motor [5]. They have built their architecture on Xilinx Virtex-5 using the Xilinx System Generator (XSG) toolbox. Since they've used XSG toolbox when implementing this control system on FPGA, we can't say exactly that their architecture is efficient in terms of memory space and maximum clock speed. On the other hand, it will not change the fact that they have done great research by comparing Direct FOC and Indirect FOC approaches. Joakim Eriksson et al. have researched a rapid prototyping system for 3-ϕ electric motor systems [6]. They concluded that a multi-axis device can be rendered with FOC using FPGA. In their work, they have verified nominal torque values, nominal

978-1-7281-9899-6/20/$31.00 ©2020 IEEE

power values and rated speed values. Besides, the PWM frequency tests have been analyzed. They have found in the simulation findings that the lower current fluctuates at large PWM frequencies. They concluded that as the PWM frequency rises slowly, the electrical motor currents are beginning to deform due to the fact that the Metal-Oxide Semiconductor Field Effect Transistor (MOSFET) may not have enough time to turn on and off entirely during the switching pulses. These problems have been quite overcome in Silicon Carbide (SiC) and Gallium Nitride (GaN) based power switches [7].

Marufuzzaman et al. have suggested a new dq PI controller focused on FPGA [8]. They argued that the new dq PI controller is the main element in increasing the overall output of the system. No matter how correct they are in their paper, there are a lot of performance criteria beyond that. Akin et al. have researched indirect control of the FPGA Induction Machine (IM) [9]. The Vector Control method has been investigated and claimed that although the DTC method is used regardless of the motor parameters, efficient feedback with DTC at low speeds can not be achieved. They therefore thought that the FOC approach would be more effective than the DTC and they have prepared FOC using the XSG toolbox on Xilinx Spartan-3.

In section II, the theory behind FOC has been mentioned. The hardware implementation and the result of the hardware implementation has been analyzed and implementation result has been discussed in section III.

II. Field Oriented Control Methodology

FOC is a Variable Frequency Drive (VFD) control methodology. In Fig. 1, based on W_{REF} value which is rotational speed command, θ value which is coming from Encoder (could be Resolver, Hall, or Sensorless mechanism), and W_{ACT} value that is calculation of speed with respect to given θ value, the circuit tries to reach W_{REF} value by applying necessary steps.

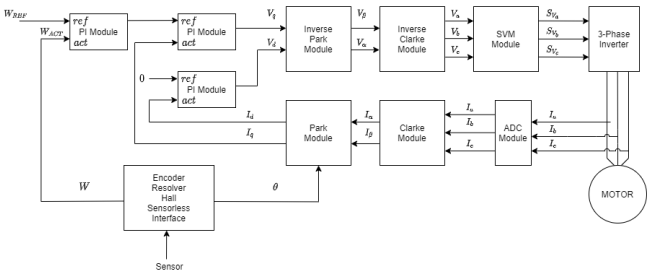

Fig. 1. FOC Flow Graph

The measurement of the motor's rotational speed is a challenge because the running motor has many disturbance factors. The main idea behind the FOC is to make more realistic observations by changing the observation frame to measure motor speed. As a result, our observation has been getting closer to real values, and driving the motor has become more stable.

In any 3-ϕ motor, the sum of 3-ϕ voltages or currents at any time should be equal to zero. By using this approach, transforming voltages or currents between the stationary frame to rotating frame or (vice versa) can be easily done. The stationary frame is called α-β frame, on the other hand, the rotational frame is called d-q frame as Fig. 2.

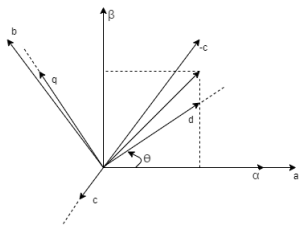

Fig. 2. α-β and d-q frame

A. Clarke & Inverse Clarke Transformation

Transforming from the 3-ϕ reference voltages or currents frame (a,b,c) to two-axis stationary frame (α,β) is called clarke or α-β transformation.

$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \frac{2}{3}\begin{bmatrix}1 & -\frac{1}{2} & -\frac{1}{2}\\ 0 & \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2}\end{bmatrix}\begin{bmatrix}a\\ b\\ c\end{bmatrix} \tag{1}$$

$$\alpha = \frac{2}{3}a - \frac{1}{3}b - \frac{1}{3}c \tag{2}$$

$$\beta = \frac{1}{\sqrt{3}}b - \frac{1}{\sqrt{3}}c \tag{3}$$

Using a + b + c = 0, the equations can be simplified as follows:

$$\alpha = a \tag{4}$$

$$\beta = \frac{a + 2b}{\sqrt{3}} \tag{5}$$

Transforming from the two-axis stationary frame (α,β) to 3-ϕ voltages or currents frame (a,b,c) is called inverse clarke transformation.

$$\begin{bmatrix}a\\ b\\ c\end{bmatrix} = \begin{bmatrix}1 & 0\\ -\frac{1}{2} & \frac{\sqrt{3}}{2}\\ -\frac{1}{2} & -\frac{\sqrt{3}}{2}\end{bmatrix}\begin{bmatrix}\alpha\\ \beta\end{bmatrix} \tag{6}$$

We can simplify matrix equation by using Eq. (4) and Eq. (5) as follows:

$$a = \alpha \tag{7}$$

$$b = -\frac{1}{2}\alpha + \frac{\sqrt{3}}{2}\beta \tag{8}$$

$$c = -\frac{1}{2}\alpha - \frac{\sqrt{3}}{2}\beta \tag{9}$$

B. Park & Inverse Park Transformation

Transforming from stationary frame (α,β) to rotating reference frame (d,q) is called park transformation.

$$\begin{aligned} d &= \alpha cos(\theta) + \beta sin(\theta) \\ q &= -\alpha sin(\theta) + \beta cos(\theta) \end{aligned} \tag{10}$$

Transforming from rotating reference (d,q) frame to stationary frame (α,β) is called park transformation.

$$\begin{aligned} \alpha &= dcos(\theta) - qsin(\theta) \\ \beta &= dsin(\theta) + qcos(\theta) \end{aligned} \tag{11}$$

C. Encoder Interface

There are four different designs to calculate θ and W values which are encoder, resolver, hall, and sensorless. The efficiency of calculation may vary according to motor types (PMSM, BLDC). While working BLDC motor type, the hall sensors provide better accuracy. On the other hand, while working PMSM motor type, the encoder sensors ensure better performance. In this study, we have chosen encoder structure to find θ and W values. The encoder structure is the quadrature encoder also known as incremental rotary encoder.

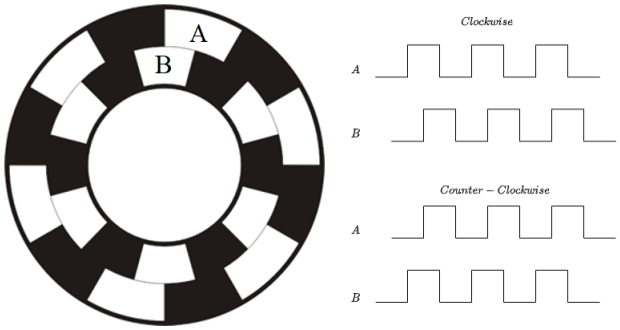

Fig. 3. Encoder Interface

As shown in Fig. 3, the direction of rotation can be easily determined. If the signal of B lagging to signal of A that means the direction of rotation is clockwise, otherwise counter-clockwise. Based on resolution of the encoder, the θ and the W can also be established.

D. PI Controller

The PI controller minimizes the error value based on input feedback and reference values. Feedback input stabilizes the unstable process due to the proportion process of PI. Since PI includes integration, PI controller output becomes an integral part of the given input. Implementation of the PI controller started with anti-windup integration, also referred to as integral windup. This feature gives the output accuracy of the PI Controller.

$$\begin{aligned} P_n &= K_p.E(n) \\ I_n &= K_i.T_s.E(n) + I_{n-1} \\ Y_n &= P_n + I_n \end{aligned} \tag{12}$$

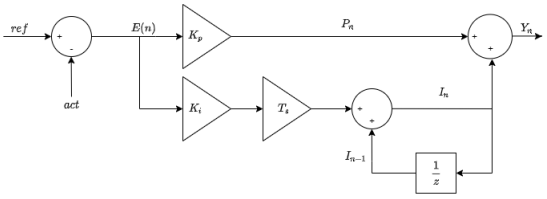

Fig. 4. PI Controller

E. Space Vector Modulation

Space Vector Modulation (SVM) is a sinusoidal wave generation technique that reduces Total Harmonic Distortion (THD) and can be used to increase the output voltage of the PWM drive. SVM has eight states that six active states, and two passive states. All of six states are driven by 3-ϕ two-level inverter. Thus, the motor has been driven. SVM is a technique that generates sine waves and feeds PWM. There's a lot of way to implement SVM. The min-max method has been used to perform SVM. Sampled voltages which has minimum value is called (V_{min}) and has maximum value is called (V_{max}). In order to calculate common voltage (V_{offset}) value as follows:

$$V_{offset} = -\frac{V_{max} - V_{min}}{2} \tag{13}$$

To skip 0-0-0 and 1-1-1 state because of increasing THD, the phase voltage value can be subtracted from common voltage value. By subtracting common voltage value to phase voltage value, it has achieved that to eliminate the third harmonic value of phase voltage.

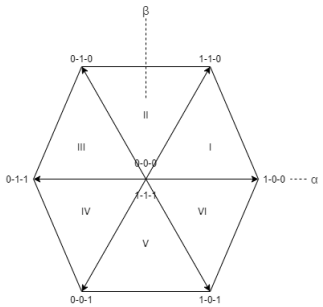

Fig. 5. SVM states

In Fig. 5, every state has max voltage value of $Vdc/\sqrt{3}$ where Vdc value is supply voltage value of 3-ϕ two-level inverter.

The theory behind the SVM is that finding V_{3H} which is third harmonic voltage and then subtracting V_{3H} from each phase voltage. The third harmonic voltage is formed as following:

$$V_{3H} = \frac{max(V_a, V_b, V_c) + min(V_a, V_b, V_c)}{2} \tag{14}$$

The calculated third harmonic voltage is subtracted to each phase voltage, space vector modulated reference voltages is found.

$$\begin{aligned} S_{V_a} &= V_a - V_{3H} \\ S_{V_b} &= V_b - V_{3H} \\ S_{V_c} &= V_c - V_{3H} \end{aligned} \tag{15}$$

III. Hardware Implementation & Results

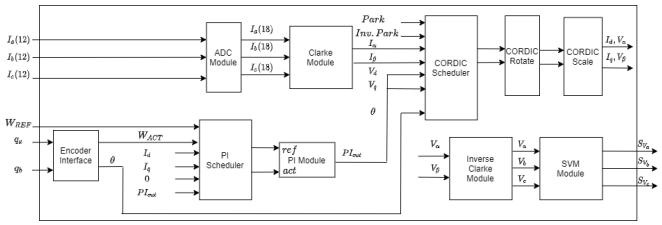

Fig. 6. Hardware Block Diagram of FOC

The Analog-to-Digital Converter (ADC) module has been prepared according to Xilinx ADC (XADC) wizard. XADC allows us to 12 bit precision, that's why we scaled up from 12 bit current values to 18 bit current values in this module. In order to construct this module, basic multiplier and simple state machine have been used. It has not been shown in the waveform for the accuracy of the test.

From the equations about clarke & inverse clarke transformation, low-level python script has been written first. Then, the result of the python script has been verified with given output vectors which are created by MATLAB. After that, Verilog module has been created. In order to be performed clarke & inverse clarke module, a simple multiplier, adder, and subtractor (MAS) module have been used.

In order to build encoder interface, the MATLAB simulation results have been taken first. Because the number of bits has been chosen as 18 for current precisions, the θ and the W precisions have been chosen same as current. The encoders have different resolution number. That's why, we have chosen a specific resolution number. According to resolution number, we have determined angle and speed factor values. The angle and speed factor values have been used to assign the θ and the W values by scaling 18-bit value. During the testbench of the FOC methodology, the encoder values have been determined, then they applied to the FOC methodology.

In order to implement park & inverse park transformation equations, the Coordinate Rotation Digital Computer (CORDIC) approach has been used. Firstly, the test vectors have been prepared by using MATLAB. Then, the low-level python script has been written according to the test vectors. After, the result of the python script has been verified with given output vectors. Following that, Verilog module has been created. While performing CORDIC algorithm, calculations have been made by applying fixed point theorem. Look Up Table (LUT)s and MAS module have been used instead of calculating the sine and cosine angle value for each time. Therefore, we have not only get rid of from the complexity of the calculation, but also from the area growth. For implementation of the park transformation, the design has been completed by taking only the reverse of the input angle (θ). Besides, the entire system has been scheduled to speed up our computations. The pipelining and resource sharing have been realized effectively.

For building PI controller module, the low-level python script has been prepared. Then, the output vectors which belong to PI module have been constructed by using MATLAB. Following that, the accuracy of the python script has been confirmed by comparing MATLAB vectors and python results. According to python results, the generic PI controller module have been generated and realized as I_d PI, I_q PI and Speed PI. In particular, the parameters of PI controller module should be initialized before running in motor control applications. For this reason, the PI controller module has been formed by using simple state machine. The K_i and K_p values have been chosen as 0.001525 and 0.097656 respectively.

The construction of SVM module, which is not very difficult theoretically, has been accomplished by the use of a simple state machine and MAS module.

For all the modules, the low-level python scripts which are 258 lines have been transformed to Verilog HDL code as 3258 lines with comment lines. The entire design have include MAS modules, CORDIC scheduler module, CORDIC rotate and CORDIC scale modules, clarke and inverse clarke modules, PI controller module, PI scheduler module, ADC module, encoder interface module, and SVM module.

The submodules which are clarke, inverse clarke, park, inverse park, ADC module, encoder interface module, PI controller and SVM module were put together to construct testbench of the FOC methodology. As shown in Fig. 1, speed PI controller is distinct from the clarke and park modules. That's why when the testbench is started, calculations of the speed PI controller can start. Because the PI controller module has initialization feature to keep a proper value before it is enabled, the PI controller output is the same as the PI controller's initial value. Keeping the PI controller's output value as its proper value avoids jerky design operation.

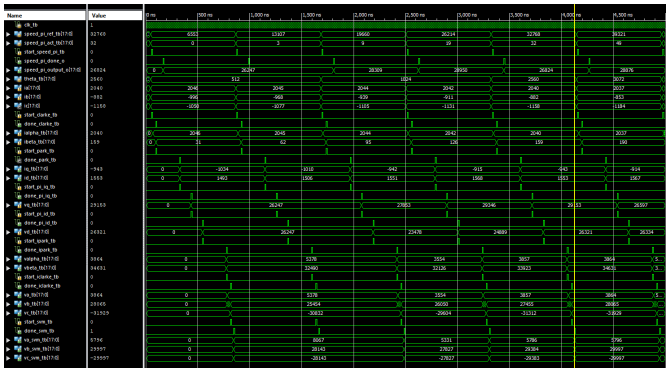

Fig. 7. Testbench Result of FOC Design

The entire design has 19 multiplication, 13 addition and subtraction, and 4 division. We have designed these operations

by using just four MAS module and binary shift logic. We have scheduled these operations by using resource sharing. Also, we have pipelined the ADC module, clarke module, and SVM module to reach higher throughput.

Based on the verification results, the ADC module have 5 cycles latency, the clarke and inverse clarke modules have 4 clock cycles latency, the park and inverse park modules have 23 clock cycles latency, the PI controller module has 11 clock cycles latency and the SVM module has 3 clock cycles latency. Encoder interface has 3 clock cycles latency but the delay of encoder interface is independent of system latency because of it is not dependent of any module.

The total latency of the FOC design is 84 clock cycles and the initiation interval of this design is one per 72 clock cycles. If the hardware clock frequency is 100 Mhz which is 10ns in terms of 1 clock cycle, the throughput value can be calculated as 1.39 Mbps. Also, maximum combinational path delay is 2.85ns and this delay is compatible with the design because of the hardware clock is higher than the maximum combinational path delay.

The comparison with papers that prefer FOC Methodology to control the 3-ϕ motor in their design is shown in Table I.

TABLE I
FOC IMPLEMENTATION COMPARISON

	FFs	LUTs	Sw. Freq.	Clk Freq.	FPGA
Kung	4174	15322	353 kHz	200 MHz	Altera Cyclone IV
Babu	5225	5514	N/A	200 MHz	Xilinx Virtex 5
Akin	1316	3172	400 kHz	50 MHz	Xilinx Spartan 3
Ours	1014	1245	1190 kHz	100 MHz	Xilinx Zynq-7020

In particular Xilinx and Altera different FPGA companies. Therefore, naming of logic units vary depending on which company you choose. There is a common index study which compares all FPGA companies in order to make them speak the same language in terms of Logic Block (LB)s [10]. Based on the common index, Altera LBs has been transformed into Xilinx LUTs as shown in Table I. Kung et al. have implemented their FOC design on Altera Cyclone IV [3]. Besides, they have implemented their design by using NIOS II processor. Akin [9] and Babu [5] have designed their FOC by using XSG. All this design approach may have caused the switching frequency to decrease. Based on Table I results, it can be said that the paper's method is superior to the other methods in terms of resource usage and maximum switching frequency.

IV. CONCLUSION

There are many advantages of a high-frequency approach in three-phase motor drive applications. Higher motor efficiency, low-cost filter, lower torque ripple, and faster control response can be among these advantages. In this study, we have offered a structure that is as fast as possible during consuming the least power. The paper's submodules (adc, clarke, inverse clarke, park, inverse park, encoder interface, PI, SVM), and testbenches that are used to verify those submodules modules have been written by using Verilog HDL. Besides, the theoretical results of those testbench modules have been written by using Python. On the other hand, this paper's work has been compared to the designs that include FOC Methodology in literature. The result of the comparison is that hardware implementation of this thesis work is provided superiority over other structures that are generated by using High-Level Synthesis (HLS) tools and HDL in terms of area and maximum switching frequency.

As for future work, this module will be applied to MATLAB Co-Sim block. Based on Co-Sim results, it can be applied to FPGA In the Loop (FIL) and it can be observed in terms of power and time. After then, it can be applied to real-time 3-ϕ motor control systems.

ACKNOWLEDGMENT

This work was supported by a TÜBİTAK BIGG project (no: 2180770).

REFERENCES

[1] J. Xu, L. Gu, Z. Ye, S. Kargarrazi, and J. Rivas-Davila, "Cascode GaN/SiC Power Device for MHz Switching," in *Proc. Applied Power Electronics Conference and Exposition (APEC)*, pp. 2780–2785, 2019.

[2] K. Shirabe, M. Swamy, J. Kang, M. Hisatsune, Y. Wu, D. Kebort, and J. Honea, "Advantages of High Frequency PWM in AC Motor Drive Applications," in *Proc. IEEE Energy Conversion Congress and Exposition (ECCE)*, pp. 2977–2984, 2012.

[3] Y. Kung, Hoang Than, Y. Lin, and L. Huang, "FPGA Based Speed Controller Design for a Ceiling Fan Motor," in *Proc. Int. Future Energy Electronics Conference (IFEEC) and ECCE Asia*, pp. 30–34, 2017.

[4] S. Suneeta, R. Srinivasan, and R. Sagar, "FPGA Based Control Method for Three Phase BLDC Motor," *Int. Journal of Electrical and Computer Engineering*, vol. 6, pp. 1434–1440, 2016.

[5] N. Babu and K. Athul, "The Field Oriented Control of Induction Motor Using FPGA," *Global Journal of Pure and Applied Mathematics*, vol. 11, pp. 1157–1170, 2015.

[6] J. Eriksson and L. Hermansen, "Rapid Prototyping: Development and Evaluation of Field Oriented Control Using LabView FPGA," Master's thesis, Mäladalen University, Sweden, 2011.

[7] K. Li, P. Evans, and M. Johnson, "SiC/GaN Power Semiconductor Devices: A Theoretical Comparison and Experimental Evaluation Under Different Switching Conditions," *IET Electrical Systems in Transportation*, vol. 8, pp. 3–11, 2018.

[8] M. Marufuzzaman, M. Reaz, and M. Ali, "FPGA Implementation of an Intelligent Current dq PI Controller for FOC PMSM Drive," in *Proc. Int. Conference on Computer Applications and Industrial Electronics (ICCAIE)*, pp. 602–605, 2010.

[9] O. Akin and I. Alan, "The Use of FPGA in Field Oriented Control of an Induction Machine," *Turkish Journal of Electrical Engineering and Computer Science*, vol. 18, pp. 943–962, 2010.

[10] O. Šćekić, "FPGA Comparative Analysis," tech. rep., University of Belgrade, 2005.