Power Electronics and Drives
3(38), No. 1, 2018 DOI: 10.2478/pead-2018-0009

Power Electronics and Drives

Space Vector Pulse Width Modulation for High-Speed Induction Motor Implemented in Nios II Softcore Processor

Research Article

Maciej Chojowski

AGH University of Science and Technology, Al. Adama Mickiewicza 30, 30-059 Kraków, Poland

Received February 15, 2018; Accepted May 31, 2018

Abstract: The purpose of the article was to present the idea of space vector pulse width modulation (SVPWM) and implementation in Nios II softcore processor. The SVPWM module was described in a classical method in hardware description language both as an independent structure and as an additional component to softcore processor. The available methods were compared, and the experiment was carried out in the laboratory to test implemented SVPWM algorithm using high-speed induction motor.

Keywords: *Induction motor • FPGA • Softcore processor*

1. Introduction

High-speed and low-power electric drives are frequently realised by DC motors. The structure of DC motor requires a commutator with brushes, and it makes the drive loud and demands replacement of worn elements. A simple construction of induction motors with lack of commutator makes three-phase drive with the induction motor highly functional. Recently, attempts have been made to replace DC motors by induction motors in household applications (Baszyński and Piróg, 2011). To control motor speed, it is possible to use an inverter with space vector pulse width modulation (SVPWM), because of cheapness and accessibility of the semiconductors. A power electronic converter can be controlled in a microprocessor unit or a field-programmable gate array (FPGA) unit or we can design our own microprocessor in FPGA called softcore processor. The FPGA unit can perform a lot of actions at exactly the same time and without any damage to functionality, but it is problematic to make user-friendly interface (it requires experience in FPGA). The SVPWM algorithm implemented in FPGAs is still the subject of scientific researches (Chaurasiya Rohit et al., 2014; Liang et al., 2016; Rashidi and Sabahi, 2013). The main aspect of the researches is to reduce the number of logical elements. Softcore processors are programmed using the C language and can be used to control the module written in hardware description language (HDL). The softcore processor with the pulse width modulation (PWM) module was already presented in Xu and Xiang (2009). The usage of the softcore processor will result in parallel implementation of the operation and the ability to easily implement the user-friendly interface. The proposed method will provide easy control of the SVPWM module in the FPGA.

* *Email: chojo@agh.edu.pl*

2. Inverter circuit

A power circuit for three-phase induction motor consists of a rectifier, DC link and inverter with three branches. A common structure of the voltage inverter is shown in Figure 1. Each branch has two insulated gate bipolar transistors (IGBT), which allow to achieve high currents and switching frequencies.

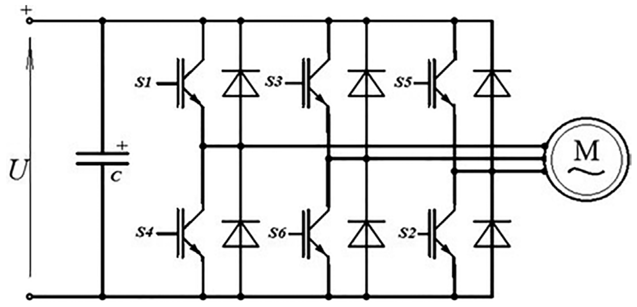

Fig. 1. Two-level DC/AC voltage source converter with an induction motor

This topology of the inverter can take eight (2^3) different configurations depending on the transistors' control signals. Transistors' logic control signals must satisfy the following logic equation:

$$S1 = \overline{S4},\ S3 = \overline{S6} \text{ and } S5 = \overline{S2} \tag{1}$$

Proper switching of transistors let us to obtain phase current of the motor close to the sinusoidal shape.

3. Space vector modulation

Three-phase voltage vectors can be presented as one vector using Clark's transformation (Baszyński and Piróg, 2011; Baszyński and Stala, 2011; Chaurasiya Rohit et al., 2014; Liang et al., 2016; Rashidi and Sabahi, 2013). Thereafter, different switching combinations of the transistors that are possible to obtain will be represented as active vectors. In $\alpha\beta$ space, the following reference vector can be generate by possible inverter states, as active (100,110,010,011,001,101) and zero vectors (111,000):

$$U_{\text{ref}} = \frac{2}{3} U e^{j(k-1)\frac{\pi}{3}} \qquad \text{for} \quad k = 1,\ 2,\ 3,\ 4,\ 5,\ 6 \tag{2}$$

$$U_{\text{ref}} = 0 \qquad \text{for} \quad k = 0 \text{ or } 7 \tag{3}$$

where *k* is the sector number, *U* the DC link voltage and *f* the frequency of the supply voltage.

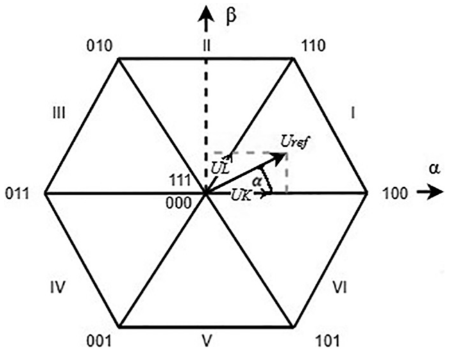

Fig. 2. The $\alpha\beta$ space with all active vectors, simple reference values and sectors

All active vectors have length equal to two-third of capacitor voltage in DC link and are shifted by 60°. The $\alpha\beta$ space is divided by vectors into six sectors as shown in Figure 2. If we want to achieve the reference vector (presented in Figure 2), we have to switch on active vectors and zero vectors for proper time. Consulting switching time equation for reference vector, U_{ref} can be described as U_{L} and U_{K} with turn on time:

$$U_{\mathrm{ref}} = U_{\mathrm{K}} + U_{\mathrm{L}} = \frac{1}{T}\left(T_a\ U_n + T_b\ U_{n+1}\right) \tag{4}$$

To calculate T_a and T_b, trigonometric formulas or matrix equation (Baszyński and Piróg, 2011; Baszyński and Stala, 2011; Chaurasiya Rohit et al., 2014; Liang et al., 2016; Rashidi and Sabahi, 2013; Ying-Yu Tzou and Hau-Jean Hsu, 1997) can be used. Results will be similar and can be described as a matrix equation:

$$\begin{bmatrix} T_a \\ T_b \end{bmatrix} = \frac{\sqrt{3}TU_{\mathrm{ref}}}{2U}\begin{bmatrix} \sin\frac{k\pi}{3} & -\cos\frac{k\pi}{3} \\ -\sin\frac{(k-1)}{3} & -\cos\frac{(k-1)}{3} \end{bmatrix}\begin{bmatrix} \cos(\alpha) \\ \sin(\alpha) \end{bmatrix} \tag{5}$$

where T is the period and α is the angle.

A method of matching auxiliary variables was presented in some scientific publications, which was used to simplify calculations (Chaurasiya Rohit et al., 2014; Liang et al., 2016; Rashidi and Sabahi, 2013). By means of the auxiliary variables, it can easily specify the sector value and the duration of the active vectors. The solutions are highly recommended by leading microcontroller manufacturers such as Texas Instrument and Freescale. On the assumption that T_a and T_b with zeroes vectors are at the beginning and the end of period, the zero vector value can be defined by the following equation (Baszyński and Piróg, 2011; Baszyński and Stala, 2011; Liang et al., 2016; Rashidi and Sabahi, 2013; Ying-Yu Tzou and Hau-Jean Hsu, 1997):

$$T_0 = T - (\ T_a + T_b) \tag{6}$$

On the basis of Figure 2, it can be presumed that the sectors will alter every 60°. The switching should be done only in one of the three control signals (S1, S3 and S5). The completed control is shown in Figure 3. It must be stated that switching pattern in figure 3 use symmetric PWM signals. The pulse signal is always symmetric with respect to the center of each PWM period.

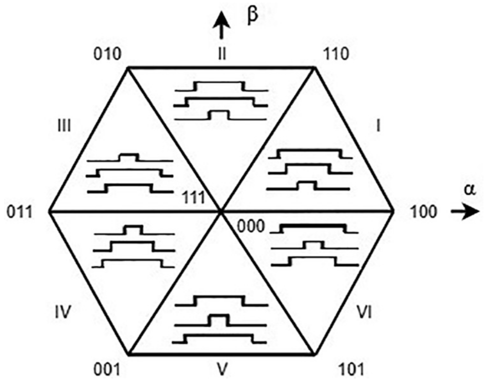

Fig. 3. Switching sequence for six sectors

4. SVPWM implementation – classical approach

The Intel FPGA unit was used to apply space vector modulation. The classic approach to implementation means that we need preparation of HDL code in the FPGA unit. The block diagram in Figure 4 depicts the idea of SVPWM realisation in the programmable logic device. To determine the times of active vectors and the development of the project, it was necessary to use the basic IEEE libraries. The product of the integer number with the real number was realised based on the number record in the Q format. An interesting way of implementation is described in Rohit et al. (2014) where the comparison of algorithm implementation by integer number and fixed-point realisation is made.

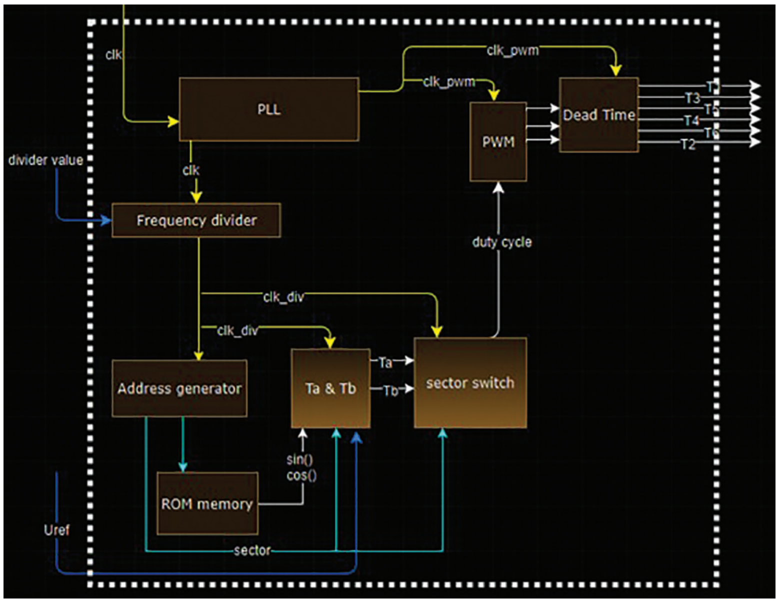

Fig. 4. Space vector realisation in FPGA – block diagram

The values of the frequency divider and the reference voltage (vector value) were provided from the outside by a user. Read only memory (ROM) was used in this project to store sine and cosine function values. After the development of the HDL code, the Signal Tap II Logic Analyzer file was added to debug and control the project in the Quartus II programme. The numerical waveforms obtained from Signal Tap II are included in Figure 5.

Figure 5 shows the durations of the active vector for the first six sectors. It can be observed that the duration of the active vector for all six sectors is identical. The implemented algorithm calculates the duration of vectors in the first sector. When the sector is changed, only active vectors are switched based on Table 1 and the time is counted in the same way. In the PWM duty factor value, the third harmonic of the signal is visible. The modulation index value ($m \approx 0.907$) for SVPWM is identical to the third harmonic injection PWM values, which confirms the results obtained in Figure 5.

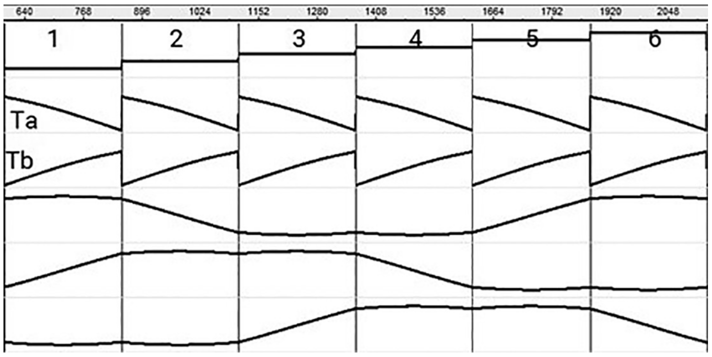

Fig. 5. T_a and T_b values during six sectors and PWM duty cycle – Signal Tap II

Table 2 summarises the percentage of used logic elements and random access memory (RAM) in the project. On the basis of the results, it can be found out that the available logic elements in this specific FPGA unit allow to implement space vector modulation and enable users to make addition components such as PID controller or interface. The only problem the users can face is the high price of the FPGA.

Table 1. Switching sequence – sector with time (Chaurasiya Rohit et al., 2014; Rashidi and Sabahi, 2013)

Sector	S1, S3 and S5	S4, S6 and S2
I	S1 = $T_0 / 2 + T_a + T_b$	S4 = $T_0 / 2$
	S3 = $T_0 / 2 + T_b$	S6 = $T_0 / 2 + T_a$
	S5 = $T_0 / 2$	S2 = $T_0 / 2 + T_a + T_b$
II	S1 = $T_0 / 2 + T_a$	S4 = $T_0 / 2 + T_b$
	S3 = $T_0 / 2 + T_a + T_b$	S6 = $T_0 / 2$
	S5 = $T_0 / 2$	S2 = $T_0 / 2 + T_a + T_b$
III	S1 = $T_0 / 2$	S4 = $T_0 / 2 + T_a + T_b$
	S3 = $T_0 / 2 + T_a + T_b$	S6 = $T_0 / 2$
	S5 = $T_0 / 2 + T_b$	S2 = $T_0 / 2 + T_a$
IV	S1 = $T_0 / 2$	S4 = $T_0 / 2 + T_a + T_b$
	S3 = $T_0 / 2 + T_a$	S6 = $T_0 / 2 + T_b$
	S5 = $T_0 / 2 + T_a + T_b$	S2 = $T_0 / 2$
V	S1 = $T_0 / 2 + T_b$	S4 = $T_0 / 2 + T_a$
	S3 = $T_0 / 2$	S6 = $T_0 / 2 + T_a + T_b$
	S5 = $T_0 / 2 + T_a + T_b$	S2 = $T_0 / 2$
VI	S1 = $T_0 / 2 + T_a + T_b$	S4 = $T_0 / 2$
	S3 = $T_0 / 2$	S6 = $T_0 / 2 + T_a + T_b$
	S5 = $T_0 / 2 + T_a$	S2 = $T_0 / 2 + T_b$

Table 2. Logic elements – classic approach

FPGA	EP3C10E144C8
Logic elements	2,277/10,320 (22%)
RAM	54,016/423,936 (13%)

5. SVPWM implementation – based on softcore processor

Owing to the complexity of the HDL, the development of a user-friendly and easy to communicate interface with the SVPWM module is extremely complicated and requires experience. A processor defined in a programmable system can be helpful in order to implement an interface that only requires knowledge of the C programming language.

The softcore processor is a microprocessor core that can be fully implemented with logic synthesis. It can be implemented by means of various devices including programmable logic, e.g. FPGA chips. Nios II is a 32-bit reduced instruction set computing (RISC) architecture processor designed by Altera, which is available in the Quartus II programming environment (Altera Corporation, 2004, 2015; Rashidi and Sabahi, 2013; Xu and Xiang, 2009). Any user-defined component can be added to the processor, e.g. SVPWM module. Components are usually the internal hardware module or an external module connected via the Avalon memory-mapped interface, which is a typical master–slave interface (Altera Corporation, 2015). It is easy to use and enables easy communication with the component.

The method of implementing SVPWM based on the softcore processor with the HDL component is proposed. The softcore processor was already used together with the PWM module as presented in Xu and Xiang (2009). The very high speed integrated circuits hardware description language (VHDL) code from the classic implementation method has been added using the Avalon memory-mapped interface to the executed processor. The Avalon interface should be configured with the HDL code according to Table 3 so that communication between the SVPWM modules and the processor can be correct.

Table 3. Avalon interface (Altera Corporation, 2015)

clock	Clock signal
resetn	Active–low reset signal
readdata	32-Bit data read from the register
writedata	32-Bit data to be stored in the register
read	Active when a read (load) transaction is to be performed
write	Active when a write (store) transaction is to be performed
byteenable	Two-bit signal that identifies which bytes are being used
chipselect	Active when the register is being read or written

The writedata signal was used to send the length of a given vector and to change the frequency of calculations in order to regulate the voltage and frequency of the motor supply, respectively. The writedata signal was entered when the write signal had a variable logic status from zero to one. The read and readdata signals were not used in the project. The illustrative module generated by the programme is shown in Figure 6.

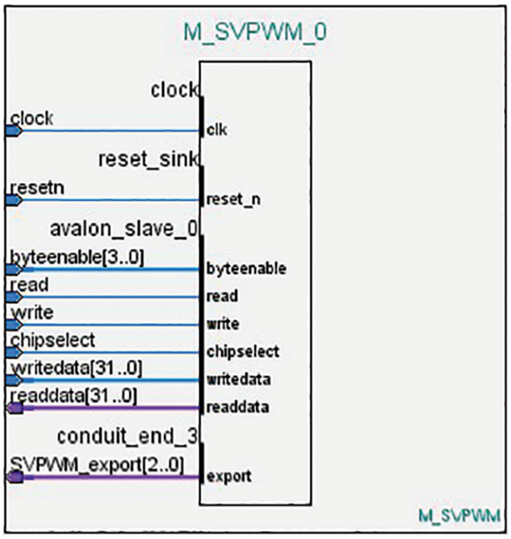

Fig. 6. SVPWM with Avalon interface – HDL component for softcore processor (Intel FPGA – Quartus II)

PWM signals (3-bit SVPWM_export signal) from the module were connected to a block that realised dead time and signal negations based on formula (1). The module and signals from the softcore processor are depicted in Figure 7.

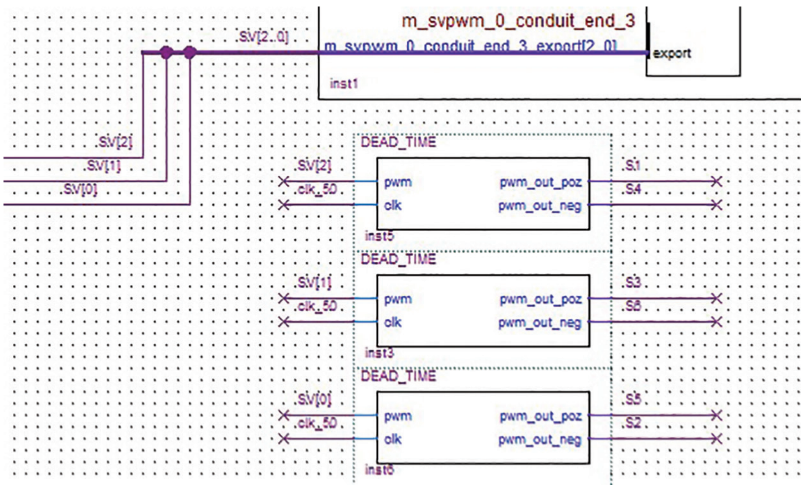

Fig. 7. Dead time realisation in VHDL (Intel FPGA – Quartus II)

After generating the softcore processor, a C code was developed. Using one function (Table 4) allows to send data to the component (Altera Corporation, 2004, 2015). The first of the function arguments was defined in the processor development stage, the component base. This is a universal number written in the form of a hexadecimal code to specify the component. The second function argument is a 32-bit data vector that determines the values of the reference vector and the frequency divider.

Table 4. Function to send data to the SVPWM module

```
IOWR_ALTERA_AVALON_PIO_DATA(SVPWM_BASE, DATA);
```

Figure 8 shows one period of PWM signals obtained during the test. The obtained results are consistent with Figure 3 (first sector). By using one function IOWR_ALTERA_AVALON_PIO_DATA, the user can communicate with the component from the processor level, which is big advantage.

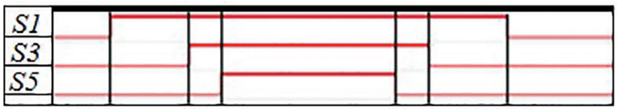

Fig. 8. Simple PWM signals for the first sector (Intel FPGA – Quartus II)

Table 5. FPGA implementation –SVPWM as an additional component to softcore processor

FPGA	EP3C10E144C8
Logic elements	5,477/10,320 (53%)
RAM	45,568/423,936 (11%)

Tables 2 and 5 show a significant increase in the logical elements used to develop the softcore processor. In the case of FPGA chips with a small number of logical elements, there may be situations in which it would not be possible by the processor but it would be necessary to replace the programmable system with an element that has a larger number of logical elements and memory.

Numerous benefits from the use of the processor such as the ease of developing the interface and easy component support from the C language level make the use of this proposed method attractive. It should also be noted that the SVPWM module is only controlled from the C language level, and all calculations are performed in parallel in the FPGA system.

6. Practical control realisation

Figure 9 shows the drive system laboratory set-up. The DC link comprises four 100 μF/500 V capacitors and was supplied from an adjustable DC power supply. The inverter contains six IGBT transistors (SK8GD126) with drivers (IR2113) and the FPGA unit (EP3C16T144). The induction motor was designed for high-frequency operation. The maximum motor speed is equal to 40,000 rpm, and the motor power is 2 kW.

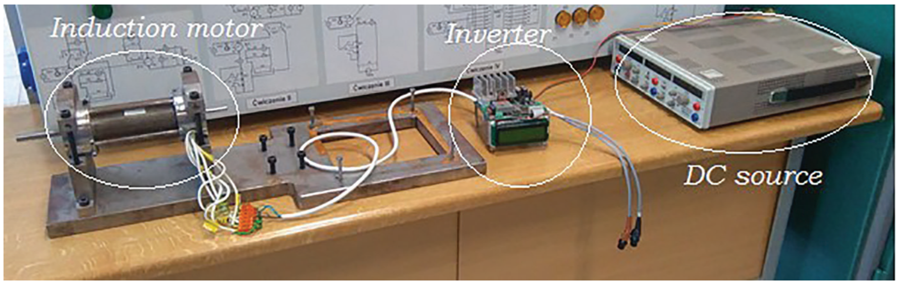

Fig. 9. High-speed induction motor with an inverter

Based on the current waveform in Figures 10 and 11, the correctness of the space vector modulation implementation was confirmed. Results were obtained for the proposed method of implementation of SVPWM using the softcore processor.

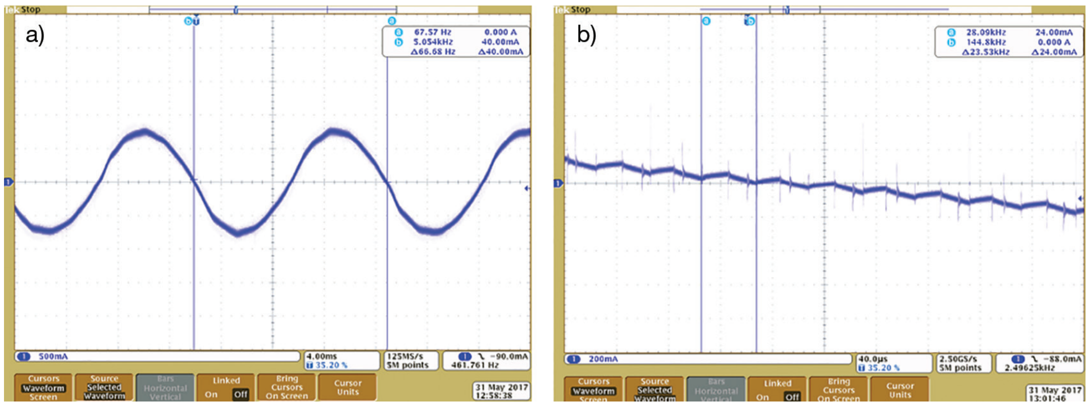

Fig. 10. The phase current of the motor current frequency 66.68 Hz (a) - change in the time base (b). The switching frequency of the transistors is 23.53 kHz

For high frequencies (Figure 11), effects of transistors' switching start to be visible in the current of the motor phase. A large strain on the motor current is to be expected with a further increase in frequency.

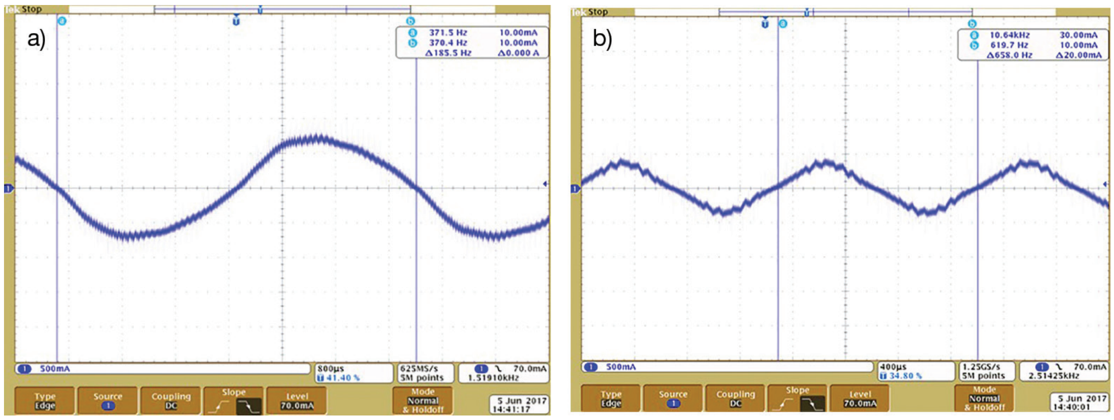

Fig. 11. The current of the motor. At a higher frequency (185.5 Hz), switching effect of transistors is visible in the motor current – deformation of a sinusoidal current signal (a). The frequency further increases from 185.5 Hz to 658 Hz (b)

In both methods, the results obtained in the laboratory are identical, but the second approach provides easier control over the SVPWM module. The presented method may be more beneficial in the implementation of large projects.

7. Conclusions

This article describes the implementation of the SVPWM control and implementations in the FPGA with Nios II processor. The theoretical considerations were confirmed, as evidenced by the obtained results. The use of the softcore processor allowed to improve the control and user-friendly interface development. The use of FPGAs makes the transfer of the simulation model to the real one significantly simplified. In addition, the processor with the component used to set the desired output values on the inverter simplifies communication and allows a simple way to control the inverter.

A user familiar with the HDL can decide for himself or herself whether to develop a control based on the classical method or to develop a softcore processor with a component and a control algorithm tailored to the needs of his or her project. The versatility of the HDL allows for easy transfer of the HDL code between FPGA systems from different manufacturers, which is an additional advantage and facilitation in case of the need to transfer the project.

References

Altera Corporation. (2004). *Introduction to Quartus II*. USA: Altera Corporation.

Altera Corporation. (2015). *Making Qsys Components*. USA: Altera Corporation.

Baszyński, M. and Piróg, S. (2011). Control of High-Speed Low-Power Induction Motor. *Electrical Review* [ISSN 0033-2097, R. 87 NR 2/2011] in polish.

Baszyński, M. and Stala, R. (2011). *Control and Modeling of Power Electronics Converters in FPGA Systems*. Cracow [ISBN: 978-83-7464-365-5] in polish.

Chaurasiya Rohit, B., Patil, M. D., Shah, D. and Kadam, A. (2014). FPGA implementation of SVPWM control technique for three phase induction motor drive using fixed point realization. In: *2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA)*, Mumbai, 2014, pp. 93–98.

Liang, X., Luo, M., Zhao, B., Yuan, Y. and Chen, Z. (2016). Research and implementation of SVPWM control algorithm based on FPGA. In: *2016 IEEE International Conference on Mechatronics and Automation*, Harbin, 2016, IEEE, pp. 22–26.

Rashidi, B. and Sabahi, M. (2013). High Performance FPGA Based Digital Space Vector PWM Three Phase Voltage Source Inverter. *International Journal of Modern Education and Computer Science*, 1, pp. 62–71.

Xu, Y. and Xiang, M. (2009). Design a new type PWM peripherals in Nios II. In: *2009 WRI World Congress on Computer Science and Information Engineering*, Los Angeles, CA, 2009, IEEE, pp. 442–446.

Ying-Yu Tzou and Hau-Jean Hsu. (1997). FPGA Realization of Space-Vector PWM Control IC for Three-Phase PWM Inverters. *IEEE Transactions on Power Electronics*, 12(6), pp. 953–963.