Real-Time Sensorless Control of the PMSM based on Genetic Algorithm, Sliding Mode Observer, and SCADA Integration

Claudiu-Ionel NICOLA, Marcel NICOLA, Maria-Cristina NIŢU,
Dumitru SACERDOŢIANU, Ancuţa-Mihaela ACIU
National Institute for Research, Development and Testing in Electrical Engineering - ICMET/Research Department, Craiova, Romania, nicolaclaudiu@icmet.ro, marcel_nicola@yahoo.com, cristinamarianitu@yahoo.com, dumitru_sacerdotianu@yahoo.com, ancutu13@yahoo.com

DOI: 10.52846/AUCEE.2021.1.04

Abstract - **This paper presents an application for real-time implementation of the Permanent Magnet Synchronous Motor (PMSM) sensorless control system and its integration into Supervisory Control And Data Acquisition (SCADA). Starting from the operating equations of the PMSM and by implementing the global Field Oriented Control (FOC) control strategy, in which the saturation of the integral component of the PI controller is prevented by using an anti-windup technique, the numerical simulations performed in Matlab/Simulink lead to good performance, which recommends the real-time implementation. For the optimal tuning of the PI speed controller of the PMSM the genetic algorithm (GA) is used. Numerical simulations are performed in order to choose the type of Digital Signal Processing (DSP) used for the real-time implementation, considering that a global criterion of successful implementation is the performance/cost ratio. Besides, the integration into SCADA provides flexibility of the control system but also the possibility of online/offline processing from the point of view of other specific requirements. Among them we mention the energy quality analysis, whose first exponent calculated also in real-time is Total Harmonic Distortion (THD). Real-time implementations are performed in Matlab/Simulink and LabVIEW programming environments. According to the trend of the last years, the use of an Internet of Things (IoT) platform for viewing the variables of the control process on the Internet plays an important role.**

Cuvinte cheie: *PMSM, FOC, algoritm genetic, observer în regim alunecător, sisteme încorporate, SCADA.*

Keywords: *PMSM, FOC, genetic algorithm, sliding mode observer, embedded systems, SCADA.*

I. INTRODUCTION

The extent of research on PMSM control with or without speed/position encoder is well-known, considering that PMSMs have a number of constructive advantages that meet the requirements of precision electrical actuators, robotics, and computer peripherals [1-5].

Among the many types of control we can mention from the PI controllers, to the adaptive, predictive and intelligent control type of controllers [6-9].

The computational intelligence has lately been providing a series of optimization algorithms which are characterized by the fact that they can be applied to the control problems under the conditions where the uncertainties are not known, and there are also unmodeled parts of the system, but the results provided by these algorithms are very good [10, 11]. Starting from the classic FOC-type control structure of a PMSM, the article presents the problem of optimally tuning the PI-type speed controller. In this respect, we start from the performance of a controller tuned in the classic manner using the Ziegler-Nichols method together with the trial and error method, and preceded by the optimization of the adjustment parameters of the PI speed controller.

Further, one of the most widely used speed observers is the SMO-type observer. The Matlab/Simulink environment is commonly used to compare the performance of these control systems. The advantages and disadvantages of various types of controllers can be thus revealed through high-precision numerical simulations. This analysis is inherently followed by the real-time implementation of the control system in embedded systems. The designer of these systems must weigh in the balance both the performance of the control systems studied by numerical simulations, and the cost of their real-time implementations in embedded systems. The real-time implementations of the PMSM sensorless control system are performed in Matlab/Simulink and LabVIEW programming environments [12-14].

Moreover, there is an inherent problem of creating a local control interface, as well as the problem of integration in the local SCADA. Thus, the control can be performed via secure communications from the Intranet/Internet. The integration into SCADA provides the transmission of signals of interest to other client systems. Thus, on the one hand, the computer server is no longer forced to perform a series of processing tasks, and on the other hand, there is the possibility of running such tasks online or offline, followed by the efficient management of global computing resources. In this sense, a separate task can run online on a client computer to monitor the quality electrical parameters of the controlled PMSM. The main parameter analyzed, but not the only one is the THD. For increased flexibility, according to the trend of the last years, the use of an IoT platform for viewing the variables of the PMSM control process on the Internet plays an important role [15-17].

The presented article is based on [20] and can be considered as a follow-up on a series of articles presented by the same authors regarding the numerical simulations of the PMSM control using various types of controllers. The tuning of the parameters of the PI controller of the outer speed loop is optimized using an optimization method based on genetic algorithm. Thus, the article presents the real-time implementation of the PMSM control and its integration into SCADA.

The rest of the paper is structured as follows: the mathematical model and the numerical simulation for the sensorless control of the PMSM is described in Section II. Section III presents the real-time implementation of the sensorless control of the PMSM and its integration into SCADA. The experimental results of the control system are presented in Section IV. Some conclusions and some ideas for the next papers are presented in the final section presents.

II. Mathematical Model Description of the PMSM Sensorless Control System

Fig. 1 presents the proposed general diagram for the PMSM sensorless control. The control system is based on the global FOC-type strategy. It can be noted that the anti-windup PI controller is fed at the input with the error between the reference speed and the speed estimated by the SMO-type observer and generates i_{qref} at the output. For the optimal tuning of the PI speed controller of the PMSM, namely parameters K_p and K_i, the genetic algorithm is used. The stator currents are acquired from the PMSM windings, and currents i_d and i_q are obtained by applying the transformation in the *d-q* reference frame. The errors between these currents and their reference values i_{qref} and i_{dref}, and i_{dref} is set to zero according to the FOC strategy, are inputs for two PI controllers which in turn generate the voltages u_d and u_q. Fig. 2 show the block diagram of the PI anti-windup speed controller.

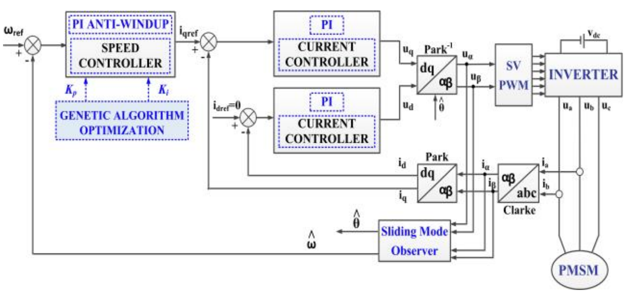

Fig. 1. The block diagram of the PMSM sensorless control system.

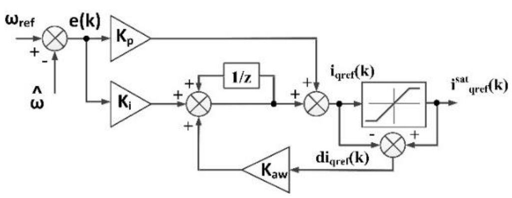

Fig. 2. The block diagram of the PI anti-windup speed controller.

The mathematical model of the PMSM in the *d-q* reference frame is the following [1], [6-8]:

$$\begin{bmatrix} u_q \\ u_d \end{bmatrix} = \begin{bmatrix} R_q + \rho L_q & \omega_e L_d \\ -\omega_e L_q & R_d + \rho L_d \end{bmatrix} \begin{bmatrix} i_q \\ i_d \end{bmatrix} + \begin{bmatrix} \omega_e \lambda_0 \\ \rho \lambda_0 \end{bmatrix} \tag{1}$$

where: u_d, u_q - stator voltages; R_d, R_q - *d*-axis and *q*-axis stator resistances; L_d, L_q - *d*-axis and *q*-axis stator inductances; i_d, i_q - *d*-axis and *q*-axis currents; ω_e - electrical angular velocity of the PMSM rotor; λ_0 - flux linkage; ρ - differential operator.

The magnetic flux of PMSM engine is expressed by the following relations:

$$\begin{aligned} \lambda_q &= L_q i_q \\ \lambda_d &= L_d i_d + \lambda_0 \end{aligned} \tag{2}$$

The following relations are obtained by using the notation T_e for the electromagnetic torque developed by the PMSM:

$$\begin{aligned} T_e &= \frac{3}{2} n_p \left(\lambda_d i_q - \lambda_q i_d \right) \\ T_e &= K_t i_q \\ T_e &= T_L + B\omega + J \frac{d\omega}{dt} \end{aligned} \tag{3}$$

where: $K_t = (3/2)(n_p \lambda_0)$ is the torque constant, B is the viscous friction coefficient, J is the moment of rotor inertia, n_p is the number of pole pairs, and T_L is the load torque.

For $L_d = L_q = L$, $R_d = R_q = R_s$, and $\omega_e = n_p \cdot \omega$, where ω is the rotor speed, the following PMSM model can be obtained:

$$\begin{pmatrix} \dot{i}_d \\ \dot{i}_q \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} -\frac{R_s}{L} & n_p \omega & 0 \\ -n_p \omega & -\frac{R_s}{L} & -\frac{n_p \lambda_0}{L} \\ 0 & \frac{K_t}{J} & -\frac{B}{J} \end{pmatrix} \begin{pmatrix} i_d \\ i_q \\ \omega \end{pmatrix} + \begin{pmatrix} \frac{u_d}{L} \\ \frac{u_q}{L} \\ -\frac{T_L}{J} \end{pmatrix} \tag{4}$$

A. Genetic Algorithm

It starts from a population of individuals (chromosomes) and creates a hierarchy of them. The number of chromosomes is usually 50, and the ranking is done according to the value of the objective function, which must be minimized. Depending on this value, but also on the values of the objective functions of the other individuals, each chromosome receives a probability of selection for reproduction. The following relation is thereby defined [10, 11]:

$$Adequate(P_{ind}) = 2 - PS + 2(PS - 1)\frac{P_{ind} - 1}{N_{ind}} \tag{5}$$

where: P_{ind} is the position of the individual, N_{ind} is the size of the population, and PS is the selection pressure, $1 \leq PS \leq 2$.

For the fittest individual, $P_{ind} = N_{ind}$ and it has the highest value of the fitness function, and, for the unfittest individual, $P_{ind} = 1$ and it has the lowest value of the fitness function.

For the tournament selection, where each individual in the current population is represented by a space proportional to the value of its evaluation function, the successive random sampling of this space of chromosome representation ensures that the best chromosomes have greater chances to be selected in a certain step than the weaker ones. Within the algorithm for the mutation function, which has the role of generating new individuals based on the random modification of the individuals in the current population, the adaptive type is selected. For replacement, which is a way of generating the new population, the fitness function will be used: number of offspring > number of parents, only the best offspring will replace parents. The structure of the GA type algorithm is shown as a graph in Fig. 3.

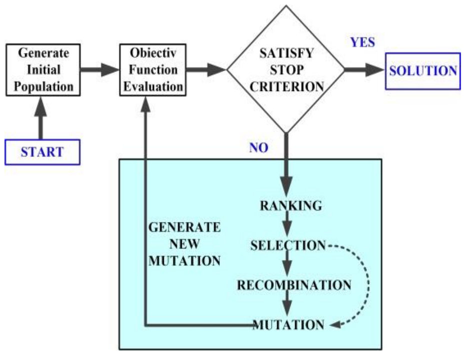

Fig. 3. The general structure of the genetic algorithm.

For each member of the population, the genetic algorithm calculates the speed error (e_ω) and the change in speed error ($s_{e\omega}$). The controller output variable is the change in the reference current (Δi_{qref}). e_ω and $s_{e\omega}$ are defined in step k as follows [10, 11]:

$$e_\omega(k) = \omega_{ref} - \omega(k) \tag{6}$$

$$se_\omega(k) = e_\omega(k) - e_\omega(k-1) \tag{7}$$

where: ω_{ref} is the reference speed.

The steps for the PMSM speed control are summarized as follows:

- The PMSM speed signal is stored in the Matlab workspace;
- The speed error is calculated and its value is updated step by step;
- The numerical representation of each parameter K_p and K_i of the PI-type speed controller is chosen;
- The crossover probability (p_c) and the mutation probability (p_m) is chosen.
- An initial population of parameters K_p and K_i is generated (yielding a random selection result);
- Δi_{qref} is generated for each member of the population C_i, i = 1,2, ...,n using the conventional PI control laws ($\Delta i_{qref}(k) = K_p e_\omega(k) + K_i se_\omega(k) T$);
- The value of the *fitness* function is assigned to each element of the population C_i, i = 1,2, ...,n.

$$p_1 = e_\omega(k) \tag{8}$$

$$p_2 = se_\omega(k) \tag{9}$$

$$F = \frac{1}{\alpha_1 p_1^2 + \alpha_2 p_2^2} \tag{10}$$

The maximum fit of the population C_i becomes C^* and the change in the control action is sent ($i_{qref}^*(k)$) to control the PMSM:

$$i_{qref}^*(k+1) = i_{qref}^*(k) + \Delta i_{qref}^*(k) \tag{11}$$

The criterion according to which the algorithm is performed is based on the integral of timed-weighted absolute error (ITAE) and has the following form:

$$f(t) = \int_0^t t|e(t)|dt \tag{12}$$

B. Sliding Mode Observer

By using the inverse Park transform, as can be noted in Fig. 1, the currents in the α-β reference frame are obtained and expressed in the following form [1], [6]:

$$\begin{aligned} \frac{di_\alpha}{dt} &= -\frac{R_s}{L} i_\alpha - \frac{1}{L} e_\alpha + \frac{1}{L} u_\alpha \\ \frac{di_\beta}{dt} &= -\frac{R_s}{L} i_\beta - \frac{1}{L} e_\beta + \frac{1}{L} u_\beta \end{aligned} \tag{13}$$

The equations of the SMO-type observer for the estimation of the PMSM rotor speed and position are based on the estimation of the back-EMF e_α and e_β.

$$\begin{aligned} \frac{d\hat{i}_\alpha}{dt} &= -\frac{R_s}{L} \hat{i}_\alpha + \frac{1}{L} u_\alpha - \frac{1}{L} kH(\hat{i}_\alpha - i_\alpha) \\ \frac{d\hat{i}_\beta}{dt} &= -\frac{R_s}{L} \hat{i}_\beta + \frac{1}{L} u_\beta - \frac{1}{L} kH(\hat{i}_\beta - i_\beta) \end{aligned} \tag{14}$$

Where the k parameter represents the observer gain, and function H is of *sigmoid* type:

$$H(x-y) = \frac{2}{1+e^{-a(x-y)}} - 1 \tag{15}$$

where: a represents a positive constant, and the *sigmoid* function indicated in relation (15) will be assigned values between -1 and 1 for a = 4.

Based on these, the estimates of the back-EMF are obtained in the following form [6]:

$$\begin{aligned} \hat{e}_\alpha &= kH(\bar{i}_\alpha) = -\lambda_0 \hat{\omega}_e \sin\theta_e \\ \hat{e}_\beta &= kH(\bar{i}_\beta) = \lambda_0 \hat{\omega}_e \cos\theta_e \end{aligned} \tag{16}$$

The PMSM rotor speed and position estimates expressed in relations (17) and (18) are obtained from the relation (16).

$$\hat{\omega}_e = \frac{\sqrt{\hat{e}_\alpha^2 + \hat{e}_\beta^2}}{\lambda_0} \tag{17}$$

$$\hat{\theta}_e(t) = \int_{t_0}^t \hat{\omega}_e(t)dt + \theta_0 \tag{18}$$

where: θ_0 is the initial electrical position of the rotor

III. Numerical Simulation

Fig. 4 shows the block diagram for implementation in Matlab/Simulink of the PMSM sensorless control system. The sensorless character of the PMSM is given by the implementation of a SMO-type observer described in previous section. The PMSM nominal parameters are presented in Table I. Fig. 5 shows the comparative time evolution of the estimated speed of the PMSM based on PI controller and PI-GA-type controller. Other quantities presented in Fig. are the load torque and electromagnetic torque, of the stator currents, and currents i_d and i_q obtained based on the numerical simulation performed in Matlab/Simulink. It can be observed that by using the genetic algorithm to optimize the tuning parameters of the PI speed controller, when applying some speed reference step signals, the system response does not show overshooting and the response time is reduced.

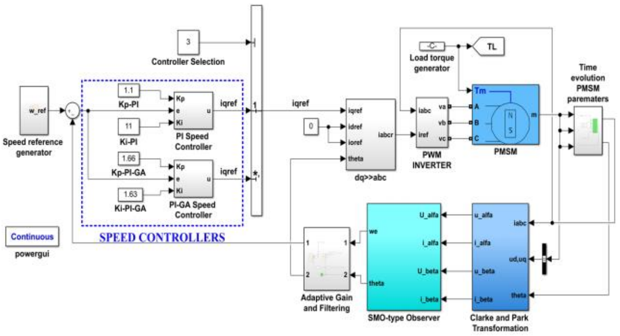

Fig. 4. The block diagram for implementation in Matlab/Simulink of the PMSM sensorless control system.

TABLE I. Nominal parameters of the PMSM

Parameter	Value	Unit
Rated voltage	24	V
Rated power	55	W
Rated speed	4000	rpm
Stator winding resistance - R_s	0.405	Ω
Stator winding inductance - L_s	0.63e-3	H
Moment of rotor inertia - J	4.6e-6	kg·m^2
Rotor friction - B	1.13e-6	N·m·s/rad
Permanent magnet flux linkage - λ_0	0.175	Wb
Pole pairs number - P	4	-

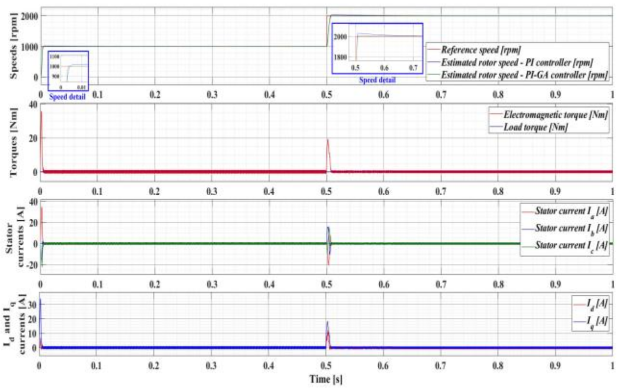

Fig. 5. Time evolution for the numerical simulation of the PMSM sensorless control system.

IV. The Architecture for the Real-Time Sensorless Control of PMSM and SCADA Integration

Fig. 6 shows the proposed architecture for the integration into SCADA of the real-time control system of a PMSM. The characteristics and operating equations of the PMSM, together with the control possibilities are presented in the previous section by means of numerical simulations in Matlab/Simulink.

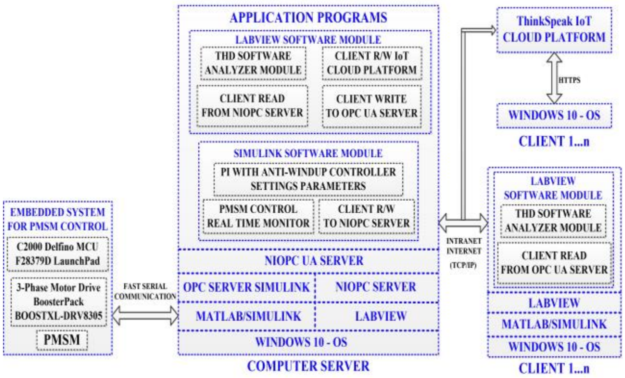

Fig. 6. The proposed general diagram for SCADA integration of the PMSM sensorless control system.

For the real-time implementation, we will use a development platform consisting of LAUNCHXL-F28379D and three-phase drive stage BOOSTXL DRV8305EVM. LAUNCHXL-F28379D use USB connected isolated XDS100v2 JTAG debug probe for real-time debug and flash programming; and contain TMS320F28379D MCU type controller with the following hardware characteristics: 200 MHz dual C28xCPUs type processor and dual CLAs, 1 MB Flash memory, 16-bit/12-bit ADCs type converter; 12-bit DACs type converter, comparators module, filters module, HRPWMs module, eCAPs module, eQEPs module, and CANs communication module. The BOOSTXL-DRV8305EVM platform is based on the DRV8305 motor gate driver and CSD18540Q5B power MOSFET. This type of module has individual DC bus and phase voltage sense as individual low-side current shunt sense for sensorless algorithms used for the control of the PMSM [17].

The block diagram of the software application for the real-time control of the PMSM is implemented in Simulink and is shown in Fig. 7. By analogy to Fig. 1, we note the implementation of the FOC type strategy, of which we mention the outer control loop for the control of speed (see Fig. 8), the inner control loop for the control of currents i_d and i_q (see Fig. 9), and the communication loop which will be required later in the software application for monitoring the control process (see Fig. 10). The running time of the two control loops is 0.05ms for the current loop, and for the speed loop it is 0.5ms.

The usual software blocks for real-time implementation are provided by the Motor Control Blockset Toolbox (Park and Clarke transforms, the sensorless observers, the anti-windup IP, etc.). By means of the Embedded Coder Support Package for TI C2000 Processors, the application program is translated into C language or machine code to be downloaded to the F28379D MCU. A code optimization is also performed, resulting in the translation from fixed point to adjustable point, leading to major improvements in the real-time control performances [17].

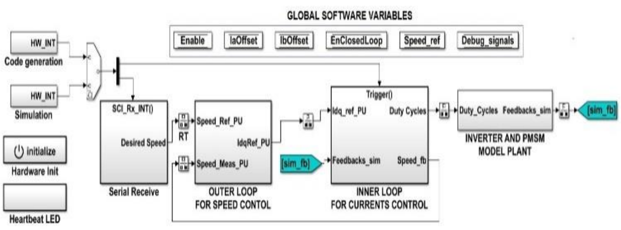

Fig. 7. The block diagram of the software application for real-time implementation in the embedded system.

The discrete time equation of the PI-type controller with anti-windup is the following:

$$i_{qref}(k) = \left[K_p + \left(K_i + di_{qref}(k) K_{aw} \right) \frac{T_s z}{z-1} \right] e(k) \quad (19)$$

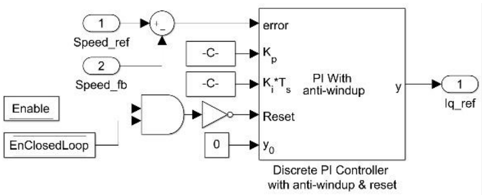

Fig. 8. The block diagram for implementation in Matlab/Simulink of the speed controller.

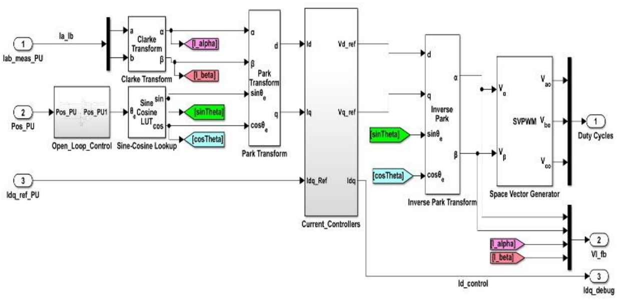

Fig. 9. The block diagram for implementation in Matlab/Simulink of the current controllers and PWM generation signals.

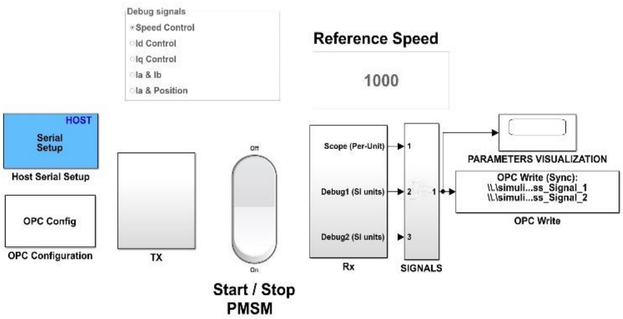

Fig. 10. The block diagram of the software application for monitoring the control process of the PMSM.

The connection between the OPC servers embedded in these software development environments provides the data stream for supervising the PMSM control between Matlab/Simulink and LabVIEW. For this purpose, we define the network-published shared variable, by which the data in the Intranet network based on Ethernet TCP/IP can be written and read. The Shared Variable Engine (EVS) mechanism uses the NI Publish-Subscribe Protocol (NI-PSP) to transfer data corresponding to the variables distributed in the data network [15, 18].

Fig. 11 shows the connection achieved between the OPC server embedded in Simulink and the "*localhost/National Instruments.Variable Engine.1*" server embedded in LabVIEW using blocks from the OPC Toolbox Simulink. The OPC Write block in SVE presented in Fig. 12 is used to perform the actual writing of the PMSM control process variables. These variables are accessible in software programs developed in LabVIEW on the computer server, but are also accessible to the embedded OPC UA server to achieve the communication with client computers connected to the Intranet/Internet.

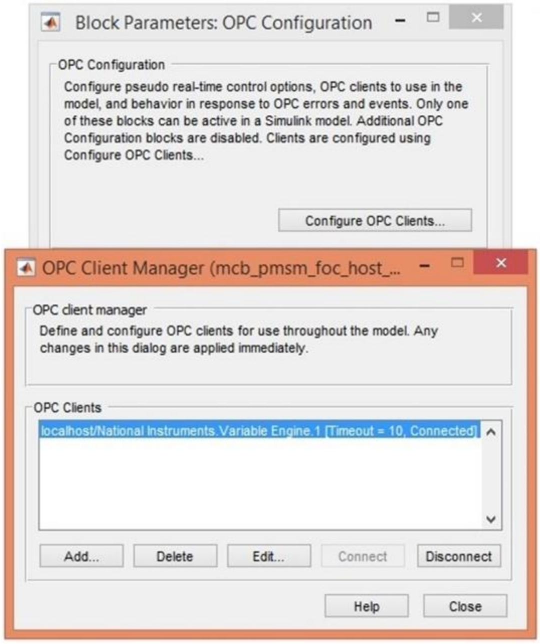

Fig. 11. Simulink OPC Server configuration.

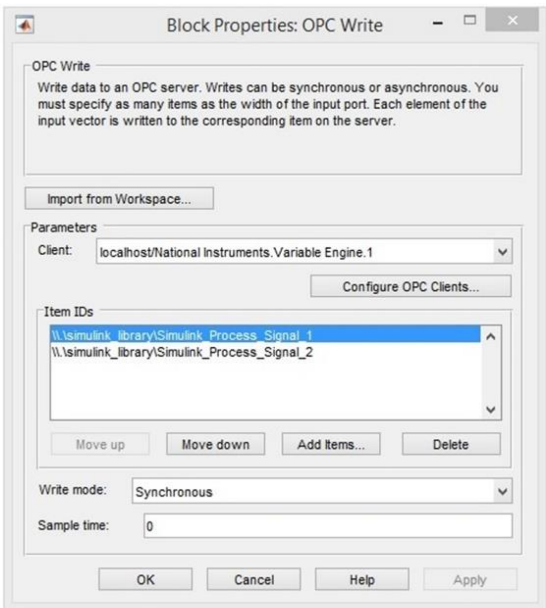

Fig. 12. Simulink OPC Server write variables of the control process.

Fig. 13 shows the LabVIEW project tree with the software applications developed on the computer server and the SVE library.

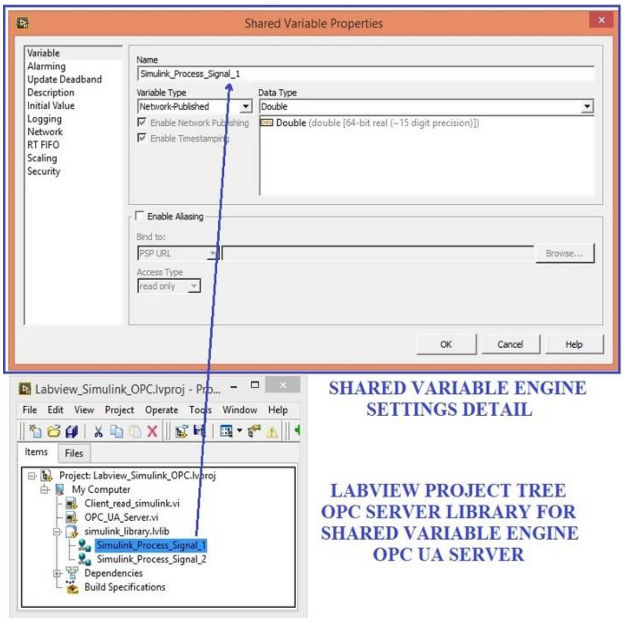

Fig. 13. LabVIEW project tree for the software applications on the computer server.

Fig. 14, Fig. 15, and Fig. 16 show the OPC UA server configuration tasks and the read/write tasks which ensure communication between the OPC UA servers on the computer server and the client software applications on the computers connected to the Internet.

Besides, a separate task can run online on a client computer to monitor the quality electrical parameters of the controlled PMSM [16].

Fig. 17 shows the implementation in a client/LabVIEW for THD analysis software module.

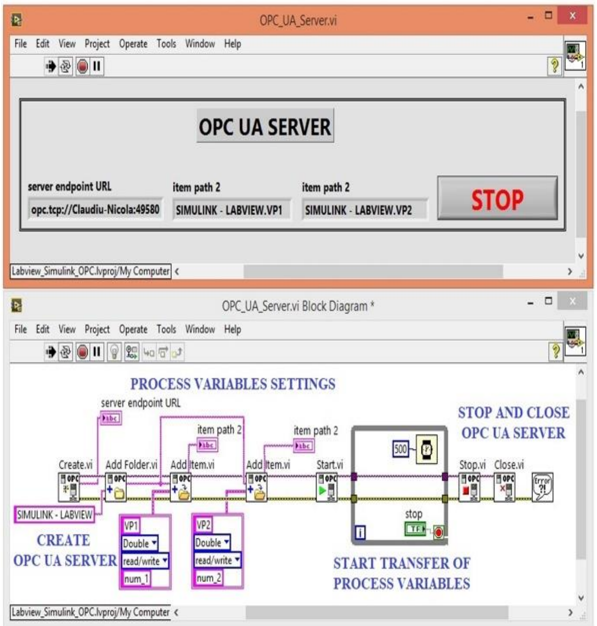

Fig. 14. OPC UA server configuration and implementation in LabVIEW.

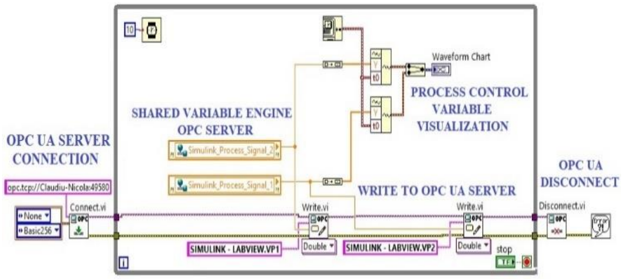

Fig. 15. Implementation in LabVIEW of the OPC UA client write.

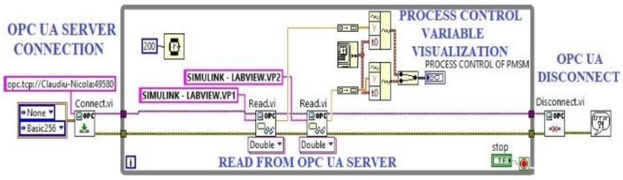

Fig. 16. Implementation in LabVIEW of the OPC UA client read.

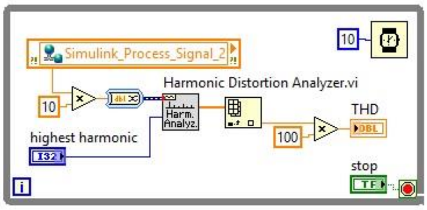

Fig. 17. Implementation in a client/LabVIEW of the task for the THD analysis.

Data from the PMSM control process can be shared in a more flexible way by using an IoT platform, which allows the data in the cloud to be viewed, analyzed and processed. The platform can be accessed from both Matlab/Simulink and LabVIEW using specific GET, POST, PUT, and DELETE commands. These commands create, write, read, or delete communication channels. These commands are HTTP requests and responses and fit into the representational state transfer (REST) architecture [19]. The configuration of a communication channel used in the presented application can be done at the address "*https://thingspeak.com/channels/1306529*" and is presented schematically as in Fig. 18.

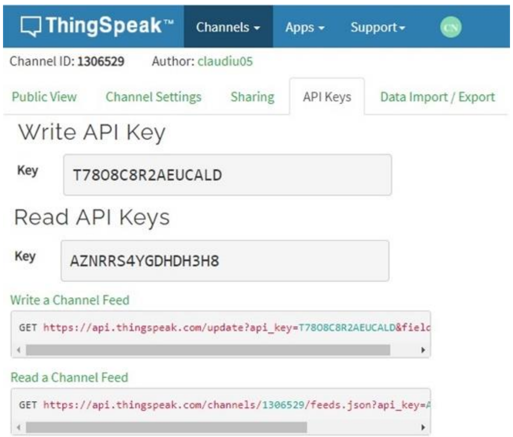

Fig. 18. Configuration settings of the IoT server for read / write channel.

A separate task implemented in LabVIEW on the computer server (see Fig. 19) writes data to this channel.

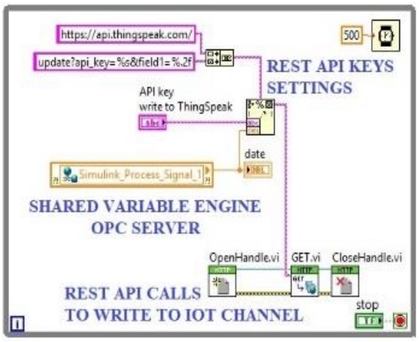

Fig. 19. Configuration settings of the IoT server for read / write channel.

V. EXPERIMENTAL RESULTS

The PMSM sensorless control system was tested in real- time in two stages. The values of the speed and the parameters of interest (currents and torque) are presented in the first stage using the interface on the host computer/computer server (see Fig.s 20-24). In the second stage, the integration into SCADA and the communication with IoT cloud platform were tested (see Fig. 25 and Fig. 26).

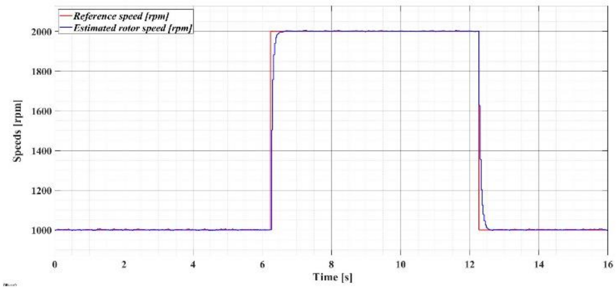

Fig. 20. Time evolution of the estimated PMSM rotor speed.

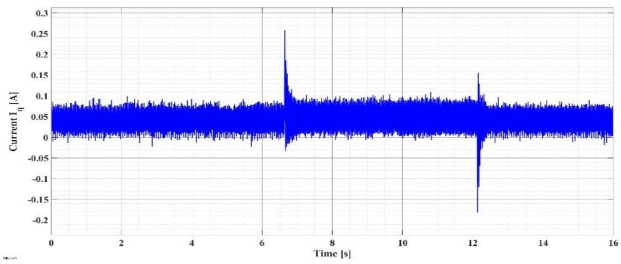

Fig. 21. Time evolution of current I_q.

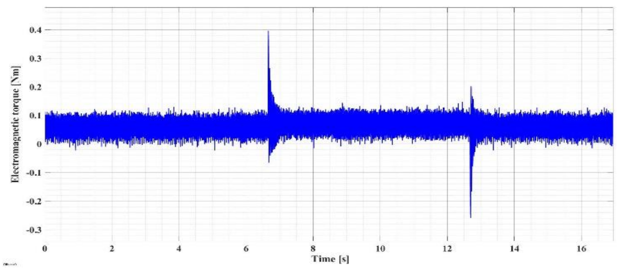

Fig. 22. Time evolution of the electromagnetic torque T_e.

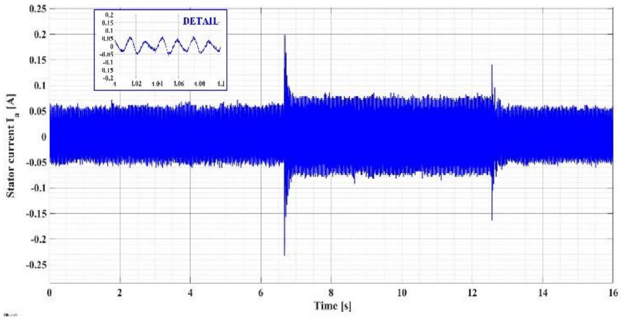

Fig. 23. Time evolution of the stator current I_a.

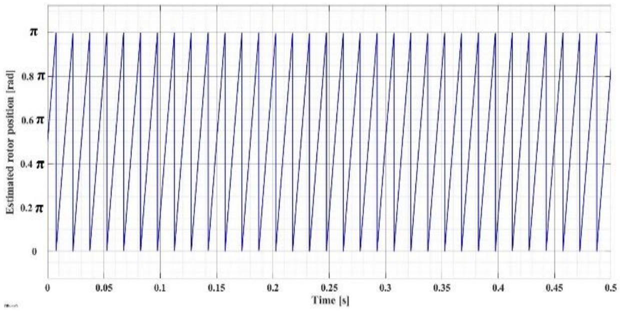

Fig. 24. Time evolution of the PMSM rotor position.

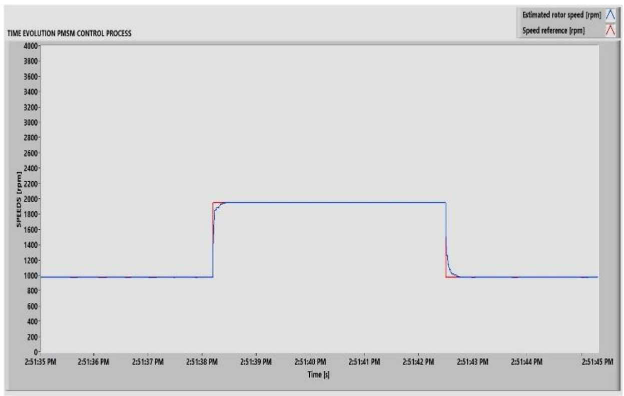

Fig. 25. Time evolution of the estimated PMSM rotor speed on OPC UA client read.

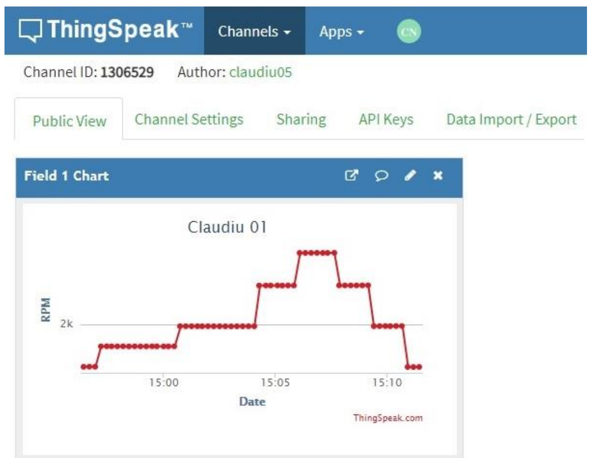

Fig. 26. Time evolution of the estimated PMSM rotor speed on IoT cloud platform.

VI. CONCLUSIONS

The article presents the mathematical models of a PMSM, of the FOC-type sensorless control, using a PI-type control with anti-windup and an SMO-type rotor speed and position observer. The tuning of the parameters of the PI-type with anti-windup speed controller is optimized by using a genetic algorithm. The article presents both the numerical simulations and the real-time implementation of the control system using a LAUNCHXL-F28379D-type development and control platform and three-phase drive stage DRV8305EVM from Texas Instruments.

OPC-type servers are used to achieve the integration into SCADA. For greater flexibility, an IoT platform is used, in order to use computers with no Simulink or LabVIEW software development modules installed. In future papers we will focus on the real-time implementation of a multi-motor application in which the control of parameters is performed locally or on the Intranet through proper integration into the local SCADA system.

ACKNOWLEDGMENT

The paper was developed with funds from the Ministry of Research and Innovation as part of the NUCLEU Program: PN 19 38 01 03 and PN 19 38 01 04.

Contribution of authors:

First author – 40%

First coauthor – 30%

Second coauthor – 10%

Third coauthor – 10%

Fourth coauthor – 10%

Received on July 31, 2021

Editorial Approval on November 20, 2021

REFERENCES

[1] V. Utkin, J. Guldner, J. Shi, Sliding mode control in electromechanical systems, second edition. Automation and Control Engineering, Taylor & Francis, 2009.

[2] B. K. Bose, Modern power electronics and AC drives, Prentice Hall, Knoxville, Tennessee, USA, 2002.

[3] H. Wang and J. Leng, “Summary on development of permanent magnet synchronous motor,” *Chinese Control And Decision Conference (CCDC)*, Shenyang, China, 2018, pp. 689-693.

[4] Z. Liu, Y. Li, and Z. Zheng, “A review of drive techniques for multiphase machines,” in CES Transactions on Electrical Machines and Systems, vol. 2, no. 2, pp. 243-251, June 2018.

[5] S. Sakunthala, R. Kiranmayi, and P. N. Mandadi, “A Review on Speed Control of Permanent Magnet Synchronous Motor Drive Using Different Control Techniques,” *International Conference on Power, Energy, Control and Transmission Systems (ICPECTS)*, Chennai, China, 2018, pp. 97-102.

[6] M. Nicola and C. I. Nicola, “Sensorless Control for PMSM Using Model Reference Adaptive Control and back -EMF Sliding Mode Observer,” *International Conference on Electromechanical and Energy Systems (SIELMEN)*, Craiova, Romania, 2019, pp. 1-7.

[7] M. Nicola and C. I. Nicola, “Sensorless Predictive Control for PMSM Using MRAS Observer,” *International Conference on Electromechanical and Energy Systems (SIELMEN)*, Craiova, Romania, 2019, pp. 1-7.

[8] M. Nicola, C. I. Nicola, and M. Duţă, “Sensorless Control of PMSM using FOC Strategy Based on Multiple ANN and Load Torque Observer,” *International Conference on Development and Application Systems (DAS)*, Suceava, Romania, 2020, pp. 32-37.

[9] M. Mutluer and O. Bilgin, “Design optimization of PMSM by particle swarm optimization and genetic algorithm,” *International Symposium on Innovations in Intelligent Systems and Applications*, Trabzon, Turkey, 2012, pp. 1-4.

[10] X. S. Yang, Nature-Inspired Metaheuristic Algorithms, 2nd Edition, Luniver Press, UK, 2010.

[11] A. Enegelbrecht, Computational intelligence an introduction, 2nd Edition, John Wiley&Sons Ltd, Chichester, West Sussex, England, 2007.

[12] C. I. Nicola, M. Nicola, A. Vintilă, and D. Sacerdoțianu, “Identification and Sensorless Control of PMSM Using FOC Strategy and Implementation in Embedded System,” *International Conference and Exhibition on Electromechanical and Energy Systems (SIELMEN)*, Craiova, Romania, 10-11 October, 2019, pp. 335-340.

[13] Z. Chen, “Permanent Magnet Synchronous Motor Parameter Test System Based on LabVIEW,” *3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, Chongqing, China, 2019, pp. 434-437.

[14] M. Sreejeth, M. Singh, P. Kumar, P. Varshney, and P. Sachdeva, “Implementation of supervisory control system for PMSM drive,” *5th India International Conference on Power Electronics (IICPE)*, Delhi, India, 2012, pp. 1-6.

[15] M. Nicola, C. I. Nicola, M. Duţă, D. Sacerdoțianu, “SCADA Systems Architecture Based on OPC and Web Servers and Integration of Applications for Industrial Process Control,” in International Journal of Control Science and Engineering, vol. 8, no. 1, pp. 13-21, April 2018.

[16] C. I. Nicola *et al*., “Quality analysis of electric energy using an interface developed in LabVIEW environment,” *International Conference on Applied and Theoretical Electricity (ICATE)*, Craiova, Romania, 2016, pp. 1-6.

[17] Motor Control Blockset™ User's Guide, Matlab and Simulink, MathWorks, Natick, MA, 2020.

[18] M. S. Mahmoud, M. Sabih, M. Elshafei, “Using OPC technology to support the study of advanced process control,” in ISA Transactions, vol. 55, pp. 155-167, March 2015.

[19] The IoT Platform with MATLAB Analytics. Available [online]: https://nl.mathworks.com/help/thingspeak/.

[20] C. I. Nicola, M. Nicola, M. C. Nițu, “Real-Time Sensorless Control of PMSM and SCADA Integration,” *International Conference on Applied and Theoretical Electricity (ICATE)*, Craiova, Romania, 2021, pp. 1-6.