Design and validation of a FPGA-based HIL simulator for minimum losses control of a PMSM

Giuseppe Galioto, Antonino Sferlazza, and Giuseppe Costantino Giaconia

University of Palermo,
Department of Engineering, viale delle scienze Ed. 9, 90128 Palermo, Italy
{giuseppe.galioto,antonino.sferlazza,costantino.giaconia}@unipa.it

Abstract. This work examines the FPGA programmable logic platforms applied to minimum losses control of a Permanent Magnet Synchronous Motor (PMSM), which represents a flexible solution for the implementation of an advanced digital control algorithm, given their intrinsic parallel structure and the capability to be directly reprogrammable in the field. In particular, design and validation of a FPGA-based Hardware-In-the-Loop (HIL) simulator is proposed, by investigating about data format, quantization and discretization effects and other issues arising during the experimental validation of a controller prototype, in order to reduce the embedded software development cycle and test control systems.
The proposed simulator has been applied to control a PMSM. Specifically, two different minimum losses control techniques have been implemented as well as a space vector modulation of a three-phases voltage source inverter. The results given in this paper show the comparison of this two different algorithms and the effectiveness of the proposed HIL simulator.

Keywords: FPGA, hardware-in-the-loop, simulations, electrical drives, PMSM

1 Introduction

HIL simulations are very important in order to validate the implementation of the controller on an electronic control unit (ECU) and they are useful tools for research where unitary tests can be conducted on the ECU without damaging the real system under test, particularly important in automotive applications [1] [2].

The use of the Field Programmable Gate Arrays (FPGAs) for HIL is sometimes required because of their performances, their computing ability and their capacity to perform parallel processing of data; they are very good candidates for real-time simulation. Indeed the FPGA can perform required calculations for real-time simulation with relatively small sample-period. However, in order to have optimum exploitation of FPGA technology, it generally requires the use of fixed-point numbers where the precision and range are defined by the designer while most DSP-based processors are equipped with floating-point unit with data format of 32 or 64 bits. It follows that an FPGA implementation is not user-friendly and usually time consuming to manage the compromise between precision and the use of FPGAs resources. Also, the increase of the data streams reduces the performances in the FPGA, justifying the importance of the arithmetic optimization of the models under study. Therefore, the minimization of the FPGA resources is very important to find the best system performance [3], [4].

This paper deals with the study of a HIL platform for minimum losses control of a PMSM ([5], [6]). Note that the minimum losses control represents an important field of study in electrical drives, also recently, with the development of new models, estimators and control algorithms dealing with it ([7], [8], [9]).

In particular, in this work the motor with its converter and sensors are modeled and implemented on Simulink, while the controller is implemented on a Xilinx board. Moreover, the Hardware Description Language (HDL) verified has been employed since it allows to test and verify Verilog and VHDL designs for FPGA, ASIC and SoC. This toolbox allows to compare the RTL with the test benches running in MATLAB or Simulink by leveraging on a co-simulation environment, tailored with an HDL simulator. Some test benches can be used with FPGA and SoC development boards to verify HDL implementations in the hardware. Therefore HDL Verifier provides tools to test and debug FPGA implementations on Xilinx and Intel boards. Using the cosimWizard command it is possible to set all the parameters related to the HDL block to be simulated such as the clock and reset signals, the output formats and the simulation time scale [10, 11].

The proposed HIL Simulator has been used for a comparison of two different minimum losses control techniques, and strengths and weaknesses of both algorithms have been highlighted. Moreover, the effectiveness of the proposed HIL simulator has been shown as well.

2 Mathematical model of PMSM

Since the main focus of the control system is to minimize the losses, the mathematical model developed should take into consideration both joule and leakage flux losses. For this reason a more complex model, with respect to a standard one, will be used. In particular the PMSM considered can be described by the following differential equations expressed in a rotating *d-q* reference frame [5]:

$$v_d = R_s i_{od} + \left(\frac{R_s + R_c}{R_c}\right)\left(\frac{d\lambda_d}{dt} - \omega_r \lambda_q\right), \tag{1a}$$

$$v_q = R_s i_{oq} + \left(\frac{R_s + R_c}{R_c}\right)\left(\frac{d\lambda_q}{dt} + \omega_r \lambda_d\right), \tag{1b}$$

where v_d and v_q are the direct and quadrature components of the stator voltage space-vector, i_{od} and i_{oq} are the direct and quadrature components of the stator current space-vector, ω_r is the rotor speed, R_s and R_c are the circuit resistances, and λ_d and λ_q are the direct and quadrature components of the flux space-vector defined as:

$$\lambda_d = L_d i_{od} + \lambda_m. \tag{2a}$$

$$\lambda_q = L_q i_{oq}. \tag{2b}$$

where λ_m is the flux produced by the permanent magnet. The mechanical equation is given by:

$$\frac{d\omega_r}{dt} = \frac{1}{J}\left(C_e - C_r - f_v \omega_r\right), \tag{3}$$

where J is the inertia moment, f_v is the viscous friction coefficient, C_r is the load torque and C_e is the electromagnetic torque produced by the motor given by:

$$C_e = \frac{3}{2} p \left(\lambda_m i_{oq} - (L_d - L_q) i_{od} i_{oq}\right), \tag{4}$$

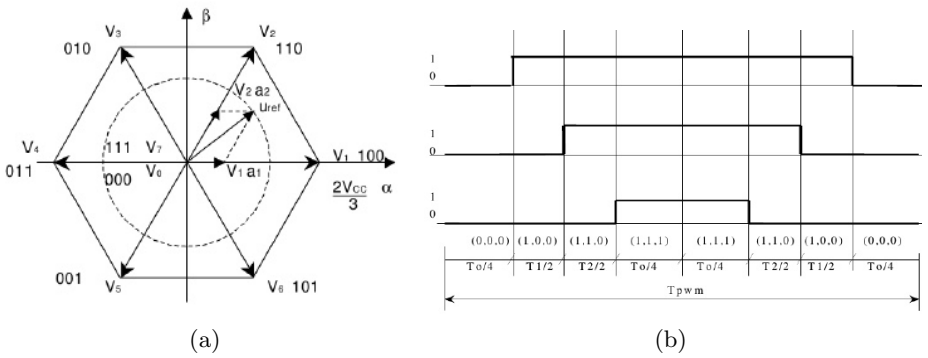
where p is the number of pole pairs and L_d and L_q are the stator inductances along direct and quadrature components respectively. Equations (1)-(4) represent the mathematical model of the system and they are implemented on Simulink in order to simulate the dynamic behavior of the system and to validate the control algorithm.

2.1 Space vector modulation of the inverter

The Space Vector Modulation (SVM) of the inverter is based on the introduction of the space vectors representative of the three-phase voltages system reproduced by the inverter and defined as:

$$\bar{V}_h = \frac{2}{3} V_{cc} \left(S_a + S_b e^{j\frac{2\pi}{3}} + S_c e^{j\frac{4\pi}{3}} \right). \tag{5}$$

where V_{cc} is the DC-voltage sourcing the inverter, S_a, S_b and S_c can assume alternatively the value 0 or 1, coherently with the logic state of the upper electronic switches of the three branches of the inverter, and $h = 0, ..., 7$ identify the eight possible states of the inverter. The eight vectors defined in (5) identify the inverter operating hex shown in Fig. 1.(a).



(a) (b)

Fig. 1. Inverter operating hex (a), firing pulses of static devices (b).

Given a reference voltage V_r contained into the operating hex, it is "*reconstructed*" by the inverter by means of a suitable sequence of space vectors, so that the average value of the output voltage equals, in the period of modulation, that of the reference voltage in the same time interval. Therefore, in terms of space vectors, it should be found that:

$$\int_0^{T_{PWM}} \left(V_r(t) - \bar{V}(t) \right) dt = 0, \tag{6}$$

$\bar{V}(\cdot)$ is the space vector generated by the inverter. Thus, if the vector $V_r(\cdot)$ is in the h-th sector of the inverter operating hex (delimited by the vectors V_h and V_{h+1}) the vector V_h is generated for a time t_h and the vector V_{h+1} for a time t_{h+1}. In general, it results that $t_h + t_{h+1} < T_{PWM}$. Therefore, in the residual time of the modulation period the null vector is applied. The right sequence of firing pulses of static devices into a modulation period T_{PWM} should be obtained as shown in Fig. 1.(b).

A easy way of implementing the above described vector modulation is to calculate the *on* and *off* instants of each device starting from the expression of the duty cycle [12], i.e. the ratio T_{ON}/T_{PWM}, in which T_{ON} is the total time in which a static device is conducting. The total computing time is reduced because it is not necessary to use trigonometric and transcendent functions for calculating the inverter operating hex sector. Indeed, as shown in [12], it is possible to define:

$$d_n = \frac{1}{2} + \frac{V_{rn} + U^*}{V_{cc}}, \quad n = a, b, c, \tag{7}$$

where:

$$U^* = -\frac{1}{2} \left(\max\{V_{r1}, V_{r2}, V_{r3}\} + \min\{V_{r1}, V_{r2}, V_{r3}\} \right).$$

Once the three duty-cycles are computed, the *on* and *off* times are computed as follows:

$$\begin{cases} T_{ONn} = \frac{T_{PWM}}{2}(1 - d_n), \\ T_{OFFn} = \frac{T_{PWM}}{2}(1 + d_n), \end{cases} \qquad n = a, b, c. \tag{8}$$

2.2 Minimum losses field oriented control of the PMSM

In this situation it is possible to adopt the field oriented control strategy to the system (1)-(4) ([12], [13]). In particular, if the current i_{od} is forced to be zero, and the two control inputs v_d and v_q are chosen, by means of a state feedback, as:

$$v_d = v_d^* - \left(\frac{R_s + R_c}{R_c}\right)\omega_r \lambda_q, \tag{9a}$$

$$v_q = v_q^* + \left(\frac{R_s + R_c}{R_c}\right)\omega_r \lambda_d, \tag{9b}$$

then the whole system can be considered as two decoupled subsystems, and the torque can be controlled by acting only on the current i_{oq}. The block diagram of the Field Oriented Control (FOC) algorithm is shown in Fig. 2.

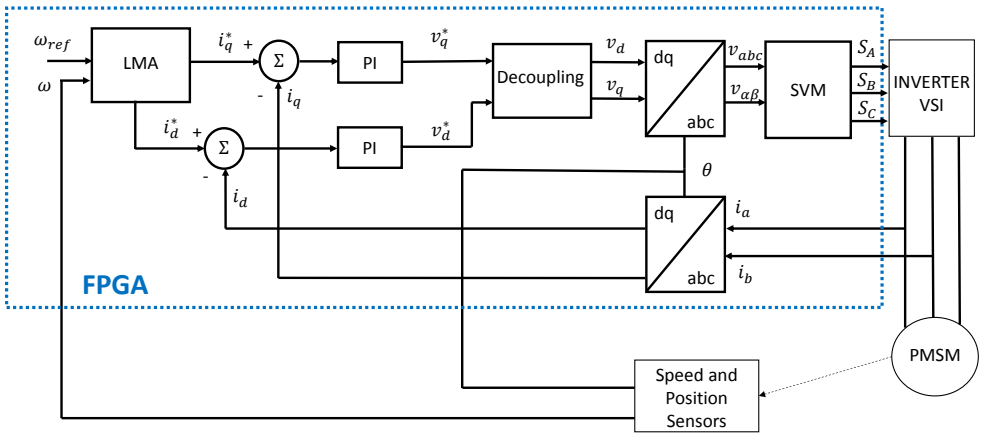


Fig. 2. Block diagram of the proposed control algorithm.

However, the main focus of the proposed application is the losses minimization for any working condition. For this reason it is important to define both electrical and mechanical losses and obtain the efficiency of the system as function of i_{od} and i_{oq}. In this work it is used the expressions given in [14] and [15].

Usually there are two methods for searching the maximum efficiency point, which correspond with the minimum losses operating condition: an open-loop model-based approach [5] and closed-loop method based on a binary search [16].

In the former model-based algorithm, the reference current value i_{od} was off-line computed so that the maximum efficiency point was reached; while the current i_{oq} was chosen

in order to respect the constraint on the required torque. This method requires a low computational effort because the minimization problem is off-line computed. However it is not robust, because the reference value of i_{od} depends on the model parameters (resistances, inductances, etc...), that are not perfectly known and can also vary during operations.

The second algorithm searches for the minimum losses point by means of a bisection procedure. In particular, fixing the search step amplitude ($d = 1\text{mA}$) and starting from an initial searching interval $[i_{odmin} = -10A,\, i_{odmax} = 10A]$, this interval is divided by two and its average value is determined as:

$$x = \frac{i_{odmin} + i_{odmax}}{2}. \tag{10}$$

The searching interval is updated using the following logic:

$$\begin{cases} i_{odmax} = x, & \text{if } W(x-d) < W(x+d), \\ i_{odmin} = x, & \text{if } W(x-d) \geq W(x+d), \end{cases} \tag{11}$$

where the expression of $W(\cdot)$ represents the total losses of the machine whose expression is:

$$\begin{aligned} W(i_{od}, i_{oq}, \omega_r) = \frac{3}{2} R_s \left(\left(i_{od} - \frac{\omega_r L_q i_{oq}}{R_c} \right)^2 + \left(i_{oq} - \frac{\omega_r (\lambda_m + L_d i_{od})}{R_c} \right)^2 \right) \\ + \frac{3\omega_r^2}{2R_c} \left((L_q i_{oq})^2 + (\lambda_m + L_d i_{od})^2 \right) \end{aligned} \tag{12}$$

The algorithm is recursively iterated until $|i_{odmin} + i_{odmax}| < 2d$, where d was the predetermined search step amplitude. Finally the value of x obtained at last iteration will be the reference value of i_{od}, which corresponds to the output of LMA block of figure 2 named i_d^*.

This second method is more robust against parameters uncertainties as will be shown in the results Section.

3 Digital implementation

The above described control algorithm was digitally implemented. In particular, for the development and synthesis of the VHDL code, Xilinx ISE Design Suite 14.7 was used, while both ISim and ModelSim were adopted for behavioural and physical simulation. The targeted system was a Zynq-7000 platform, that is an all programmable SoC, and in the case of ModelSim simulation it was necessary to use the following library:

```
library ieee; use ieee.fixed_pkg.all;
```

The chosen representation for all variables and signals has been the signed [16.16] format, resulted as a good compromise between precision and computational efficiency.

The PI controllers are obtained by implementing the following discrete equation:

$$y(k) = y(k-1) + K_p e(k) + K_i T_s e(k) \tag{13}$$

where k is the k-th sampling instant and T_s represents the sampling period. System behavior was analyzed in fixed-point [16.16] format and regulators calibration were kept unchanged for the continuous and discrete case; while sampling time of the integrator has been set at 10 μs (100 kHz), considering that the converter's switching frequency was fixed at 10 kHz.

The equations that describe vector modulation are intrinsically digital and therefore required no further manipulation in order to be digitally implemented. It was necessary to implement a up/down counter for determining the switching instants of the inverter.

This counter should give a triangular waveform with a 10 kHz period and count, in the up/down case, up to a value equal to $T_{PWM}/2$. The modulation period can be expressed in arbitrary units and in this case, for convenience, we have chosen to express it in μs. Since $T_{PWM} = 100\mu$s, the counter counts from 0 to 50 up/down at a frequency equal to 10 kHz. Of course the counter clock signal should have a higher frequency and it was set to 1 MHz, easily attainable with the adopted FPGA, starting from the system clock and coding a suitable frequency divider.

For obtaining the VHDL code of the model-based Losses Minimization Algorithm (LMA) it was necessary to make changes of the numerical format used. Only for this block, in fact, the operations were carried out using a 64-bit format [32.32], and then truncating the obtained output in order to have a fixed-point [16.16] format value of the reference current i_{od}. This has been necessary because a low accuracy of the algorithm was experimentally observed. Furthermore a study has been carried out in order to take into account for relatively strong deviations of the captured speed and position signals due to possible errors introduced by the analog to digital conversion, observing that a decrease of the effective ADC resolution down to 9 bit can be easily tolerated by the control algorithm. For the implementation of the binary-search algorithm, the fixed-point [16.16] format has been used and, as shown in Fig. 3, a state machine was considered.

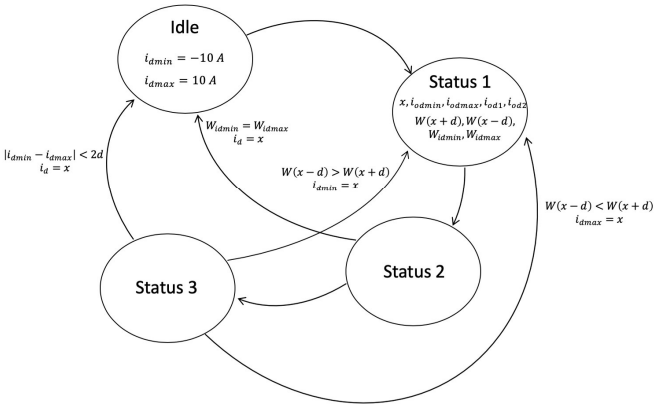
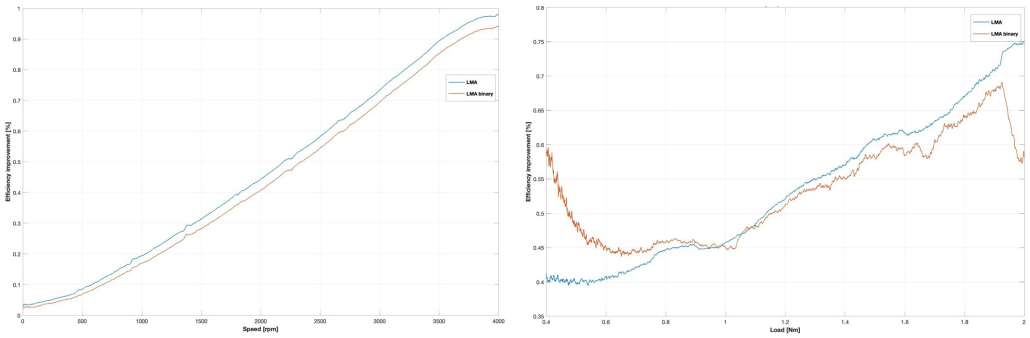


Fig. 3. States diagram of the LMA algorithm.

4 Results

The results obtained have been shown in Fig. 5 and Tab.s 1-3. In particular, they take as a reference the efficiency of classical FOC method and depict the percentage improvements attainable with the two minimization algorithms. Two tests have been carried out, the first implementing a variable speed waveform, and a second test with a variable torque profile. The results of efficiency improvements are shown in Fig. 5, which compares the model-based and binary-search algorithms when nominal parameters are used. In this case the model-based algorithm gave better performances only because a good knowledge of the model was available. However, if the parameters are detuned, the efficiency improvement of the binary search algorithm overcomes the one of the model-based algorithm, since it is based on the measurements of the state variable obtained directly from the real model, and the



(a) (b)

Fig. 4. Comparison between model-based and binary-search algorithms as function of the speed (a) and of the torque (b).

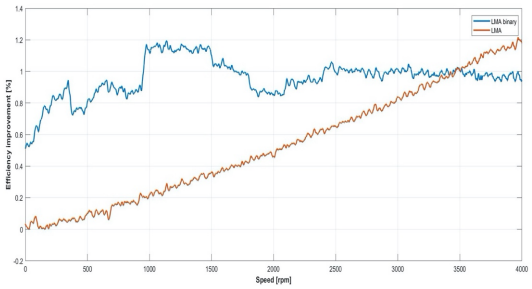


Fig. 5. Comparison between model-based and binary-search algorithms as function of the speed when the nominal parameters are detuned of 10%.

Table 1. Device utilization summary for the classical FOC algorithm

Platform: Xilinx Kintex-7 FPGA - XC7K160T (28 nm technology)

Logic utilization	Used	Available	Utilization
Number of slice registers	3325	202800	1%
Number of slice LUTs	23816	101400	23%
Number of fully used LUT-FF pairs	1219	25922	4%
Number of bonded IOBs	197	285	64%
Number of BUFG/BUFGCTRLs	5	32	15%
Number of DSP48E1s	118	600	19%

Minimum period: 101.558 ns (maximum frequency: 9.847 MHz)
Minimum input arrival before clock: 24.581 ns
Maximum output required after clock: 2.532 ns

Table 2. Device utilization summary for the model-based algorithm

Platform: Xilinx Kintex-7 FPGA - XC7K160T (28 nm technology)

Logic utilization	Used	Available	Utilization
Number of slice registers	4049	202800	1%
Number of slice LUTs	75140	101400	74%
Number of fully used LUT-FF pairs	1545	77644	1%
Number of bonded IOBs	197	285	69%
Number of BUFG/BUFGCTRLs	5	32	15%
Number of DSP48E1s	183	600	30%
Minimum period: 225.410 ns (maximum frequency: 4.436 MHz)			
Minimum input arrival before clock: 24.579 ns			
Maximum output required after clock: 2.271 ns			

Table 3. Device utilization summary for the binary-search algorithm

Platform: Xilinx Kintex-7 FPGA - XC7K160T (28 nm technology)

Logic utilization	Used	Available	Utilization
Number of slice registers	5927	202800	2%
Number of slice LUTs	59430	101400	58%
Number of fully used LUT-FF pairs	1705	63652	2%
Number of bonded IOBs	197	400	49%
Number of BUFG/BUFGCTRLs	7	32	21%
Number of DSP48E1s	248	600	41%
Minimum period: 101.558 ns (maximum frequency: 9.847 MHz)			
Minimum input arrival before clock: 24.581 ns			
Maximum output required after clock: 2.586 ns			

parameter uncertainties did not affect the efficiency improvement. Finally, Tab.s 1-3 gives a summary of the device utilization for three cases: classical FOC algorithm (Tab. 1), model-based algorithm (Tab. 2), binary-search algorithm (Tab. 3). From this plot it is evident that the binary-search algorithm requires a lower computational effort as expected, while remaining well within the capabilities of the adopted FPGA.

References

1. Fathy, H.K., Filipi, Z.S., Hagena, J., Stein, J.L.: Review of hardware-in-the-loop simulation and its prospects in the automotive area. In: Modeling and simulation for military applications. Volume 6228., International Society for Optics and Photonics (2006) 62280E–1 – 62280E–20
2. Bouscayrol, A.: Different types of hardware-in-the-loop simulation for electric drives. In: 2008 IEEE International Symposium on Industrial Electronics, IEEE (2008) 2146–2151
3. Tavana, N.R., Dinavahi, V.: A general framework for FPGA-based real-time emulation of electrical machines for HIL applications. IEEE Transactions on Industrial Electronics **62**(4) (2014) 2041–2053
4. Ould-Bachir, T., Dufour, C., Bélanger, J., Mahseredjian, J., David, J.P.: Effective floating-point calculation engines intended for the FPGA-based HIL simulation. In: 2012 IEEE International Symposium on Industrial Electronics, IEEE (2012) 1363–1368
5. Morimoto, S., Tong, Y., Takeda, Y., Hirasa, T.: Loss minimization control of permanent magnet synchronous motor drives. IEEE Transactions on industrial electronics **41**(5) (1994) 511–517
6. Lee, J., Nam, K., Choi, S., Kwon, S.: Loss minimizing control of PMSM with the use of polynomial approximations. In: 2008 IEEE Industry Applications Society Annual Meeting, IEEE (2008) 1–9

7. Accetta, A., Cirrincione, M., Pucci, M., Sferlazza, A.: State space-vector model of linear induction motors including end-effects and iron losses part I: theoretical analysis. IEEE Transactions on Industry Applications **56**(1) (2019) 235–244
8. Alonge, F., Cirrincione, M., Pucci, M., Sferlazza, A.: A nonlinear observer for rotor flux estimation of induction motor considering the estimated magnetization characteristic. IEEE Transactions on Industry Applications **53**(6) (2017) 5952–5965
9. Accetta, A., Alonge, F., Cirrincione, M., D'Ippolito, F., Pucci, M., Rabbeni, R., Sferlazza, A.: Robust control for high performance induction motor drives based on partial state-feedback linearization. IEEE Transactions on Industry Applications **55**(1) (2018) 490–503
10. MathWorks: Hdl coder - https://www.mathworks.com/products/hdl-coder.html
11. MathWorks: Hdl verifier - https://www.mathworks.com/products/hdl-verifier.html
12. Vas, P.: Sensorless vector and direct torque control. Oxford Univ. Press (1998)
13. Leonhard, W.: Control of electrical drives. Springer Science & Business Media (2001)
14. Uddin, M.N., Zou, H., Azevedo, F.: Online loss-minimization-based adaptive flux observer for direct torque and flux control of PMSM drive. IEEE Transactions on Industry Applications **52**(1) (2015) 425–431
15. Sato, D., Itoh, J.i.: Total loss comparison of inverter circuit topologies with interior permanent magnet synchronous motor drive system. In: 2013 IEEE ECCE Asia Downunder, IEEE (2013) 537–543
16. Cavallaro, C., Di Tommaso, A.O., Miceli, R., Raciti, A., Galluzzo, G.R., Trapanese, M.: Efficiency enhancement of permanent-magnet synchronous motor drives by online loss minimization approaches. IEEE Transactions on Industrial Electronics **52**(4) (2005) 1153–1160