Hybrid Electric Vehicle Experimental Model with CAN Network Real Time Control

Gheorghe LIVINȚ, Vasile HORGA, Daniel STICEA, Marcel RĂȚOI, Mihai ALBU
Gheorghe Asachi Technical University of Iaşi, Bvd. Dimitrie Mangeron, nr. 23, RO-700050, Iaşi,
glivint@tuiasi.ro, horga@tuiasi.ro, danisticea@yahoo.com ratoi@tuiasi.ro, albu@tuiasi.ro

***Abstract* — In this paper an experimental model with a distributed control system of a hybrid electrical vehicle is presented. A communication CAN network of high speed (1 Mbps) assures a distributed control of the all components. The modeling and the control of different operating regimes are realized on an experimental test-bench of a hybrid electrical vehicle. The experimental results concerning the variations of the mains variables (currents, torques, speeds) are presented.**

***Index Terms* — CAN network, experimental model, hybrid electrical vehicle, real time control**

I. INTRODUCTION

Comparing to a classic vehicle, the hybrid one is more complex (about 25% complexity on hardware and more than double complexity on software). In order to obtain the maximum efficiency of the hybrid vehicle the main control strategy is to select the propulsion force (engine or electric motor) depending on the load. The engine has a low efficiency at low load, for transient regimes and for idling. For the full loads and high speed the engine has the maximum efficiency.

The control strategy for the hybrid vehicle is trying to avoid these regimes by using the control algorithms to manage the energy sources in order to minimize the fuel consumption and the emissions [5], [10], [12].

Thus, a hybrid vehicle, with a proper control, will consume less fuel (about a half of a classic vehicle with similar engine characteristics). In this case the vehicle autonomy will be double and the emissions will be lower because of the transients and idling regimes elimination.

II. EXPERIMENTAL MODEL OF A HYBRID ELECTRIC VEHICLE

The structure of the experimental model of the hybrid vehicle is presented in Fig. 1. The model includes the two power propulsion (engine – E, and the electric motor/generator M/G) which allow the energetically optimization by implementing the real time control algorithms. The model has no wheels and the longitudinal characteristics emulation is realized with a corresponding load system. The engine is a diesel F8Q of 1.9l capacity and 64 HP. The Electronic Unit Control (ECU) is a Lucas DCN R04080012J-80759M. The coupling with the motor/generator system is assured by a clutch, a gearbox and a belt transmission.

The electric machine is a squirrel cage asynchronous machine (15kW, 380V, 30.5A, 50Hz, 2940 rpm) supplied by a PWM inverter implemented with IGBT modules (SKM200GB122D). The motor is supplied by 26 batteries (12V/45Ah). The hardware structure of the motor/ generator system is presented in Fig. 2.

The hardware resources assured by the control system eZdsp 2808 permit the implementation of the local dynamic control algorithms and for a CAN communication network, necessary for the distributed control used on the hybrid electric vehicle.

The peripheral elements (8 ePWM channels, 2x8 AD channels with a resolution of 12 bits, incremental transducer interface eQEP) and the specific peripheral for the communication assure the necessary resources for the power converters command and for the signal acquisition in system. For the command and state signal conditioning it was designed and realized an interface module.

III. THE EMULATION OF THE LONGITUDINAL DYNAMICS CHARACTERISTICS OF THE VEHICLE

The longitudinal dynamics characteristics of the vehicle are emulated with an electric machine with torque control (Fig.3). As a mechanical load emulator, the electric machine operates both in motor and generator regimes. An asynchronous machine with vector control technique assures a good dynamic for torque.

This asynchronous machine with parameters (15KW, 12A, 380V) is supplied by a SINAMICS S120 converter from Siemens which contains a PWM rectifier, a voltage dc link and a PWM inverter [6]. This converter assures a sinusoidal current at the network interface and the possibility to recover into the network the electric energy given by the electric machine when it operates in generator regime.

The main objective is to emulate the static, dynamics and operating characteristics of the drive line.

The power demand for the vehicle driving at a constant speed and on a flat road [5], [12] can be expressed as

$$P_e = \frac{v}{1000\eta_{t,e}}\left(m_v g f_r + \frac{1}{2}\rho_a C_D A_f v^2 + m_v g i\right)\ [kW] \quad (1)$$

where: m_v - the vehicle mass, f_r - rolling resistance coefficient, ρ_a - air density, C_D - aerodynamic drag coefficient, A_f - front area, $\eta_{t,e}$ - transmission efficiency from engine to drive wheels, i - the road grade. The average power requirement for the vehicle driving in a stop-and-go pattern, in a drive cycle can be calculated by

$$P_{ave} = \frac{1}{T}\int_0^T \left(m_v g f_r v + \frac{1}{2}\rho_a C_D A_f v^3 + \delta m_v v \frac{dv}{dt}\right)dt \quad (2)$$

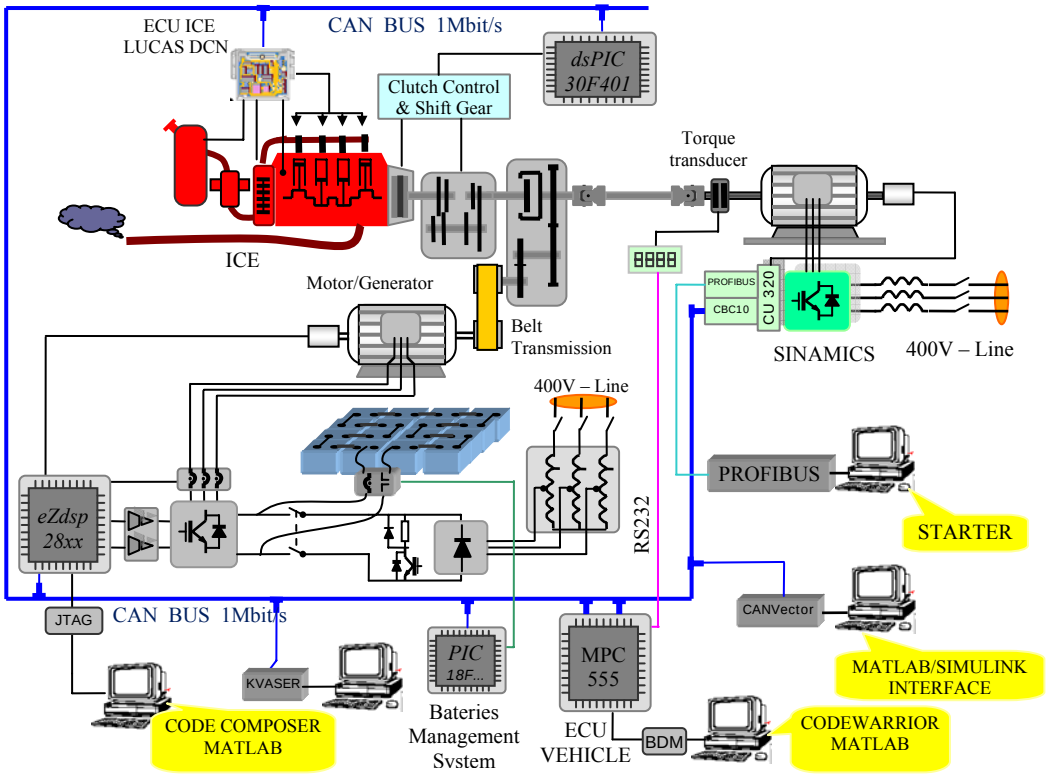

Figure 1. The structure of the experimental model of the hybrid electric vehicle.

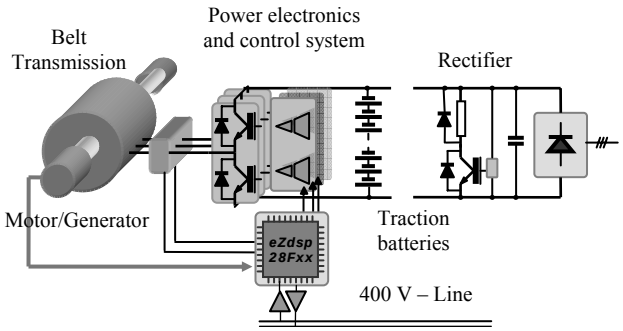

Figure 2. Electric motor/generator system.

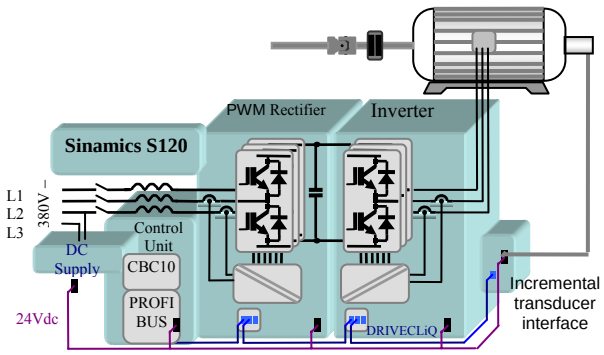

Figure 3. Emulation system of the longitudinal dynamics characteristics of the vehicle.

IV. THE DISTRIBUTED SYSTEM OF THE REAL-TIME CONTROL OF THE HYBRID ELECTRIC VEHICLE MODEL

The coordinated control of the sub-systems of the parallel hybrid vehicle can be realized with a hierarchical structure, [5], [8]. Its main element is the Electronic Control Unit vehicle of the vehicle (ECU vehicle) which supervises and coordinates the whole systems.

It has to monitor permanently the driver demands, the motion conditions and the state of the sub-systems in order to estimate the optimum topology of the whole system and to assure minimum fuel consumption at high running performances. The main system must to assure the maneuverability demanded by the driver in any running conditions. These supervising and coordinating tasks are realized by a control structure that includes both state automata elements and dynamic control elements corresponding to each state [8], [9]. The dynamic control of each sub-system is realized by every local control system. The dynamic control is integrated at the level of the coordinating system only when it is necessary a smooth transition between states or for a dynamic change into a state with more than a sub-system (starting engine with the electric machine).

The optimization of the performances objectives is realized logically by the state automata. The optimum operating state is determined by the coordinating and supervising system based on the analysis of the centralized data.

The state machine design is achieved in three stages:

- the identification of the all possible operating modes of the vehicle,
- the evaluation of the all possible transitions between the operating modes,
- the arbitration of the priorities between the concurrent transitions.

For the first stage it is realized a list with the possible operating modes for each sub-system. For example, for the engine the possible operating modes are running engine and stop engine.

After the identification of the all possible operating modes of each sub-system, it is generated a set of all the possible combinations of the operating modes for the vehicle.

Due to the complexity of the real time control for a parallel hybrid electric vehicle it is necessary to integrate all the elements in a high speed CAN communication network (1Mbps) to assure the distributed control of all resources [1] – [4],[7], [11]. The experimental model uses a CANopen network with four slave nodes and one master node. The master node is implemented on phyCORE-mpc555 system and assures the network management and supervises the nodes control connected by NMT services, the nodes operating states, the emergency messages analysis and the modifications appeared into the communication network.

The first CANopen slave node, at an inferior level, is dedicated for the motor/generator system and includes the speed control loop for the vehicle electrical propulsion.

The second slave node is used to take over the torque data given on the RS-232 serial line by the DTR torque transducer and to convert the data for the proper utilization on the CANopen network.

The third slave node of the CANopen network is used for the emulation system for the longitudinal dynamics characteristics of the vehicle, implemented with the asynchronous motor and the SINAMICS S120 converter.

The fourth slave node of the CANopen network is the system of automatic gear shift, which involves control of clutch and gear. Control is achieved with a dsPIC-30F4011 numerical system.

The CAN protocol utilizes versatile message identifiers that can be mapped to specific control information categories. With predefined priority of the communication message, non-destructive bit-wise arbitration with error detection signaling, the CAN protocol supports distributed real-time control in vehicles applications with a very high level of security [10].

The content of a message is named by an *identifier*.

The *identifier* describes the meaning of the date, but not indicates the destination of the message. All nodes in the network are able to decide by *message filtering* whether the data is to be accepted. If two or more nodes attempt to transmit at the same time, a non-destructive arbitration technique guarantees the messages transmission in order of priority and that no messages are lost.

It is guaranteed that a message is simultaneously accepted by all nodes of a CAN network. When a receiver detects an error in the last bit that cares about it will send an error frame and the transmitter will retransmit the message.

The CAN network provides standardized communication objects for real-time data (PDO – Process Data Objects), configuration data (SDO – Service Data objects), and special functions (Emergency message), network management data (NMT message, Error control).

Service Data Object (SDO) supports the mandatory OD (Object Dictionary) entries, slave support for the next slave services: Reset_Node, Enter_Preoperational_State, Start_Remote_Node, Stop_Remote_Node, Reset Communication, COB (Communication Data Object).

For the software design it was in attention the modularity and a scalar structure of the final product that can be easy configured for the automation necessities of the communication node. For this the CANopen stack was structured in two modules:

- Module I, dependent on the hardware resources of the numerical system,

- Module II, specific for the application, independent on the hardware resources. To pass the product on other numerical systems it is enough to rewrite the first module.

The functional structure of the slave CANopen software is presented in Fig. 4.

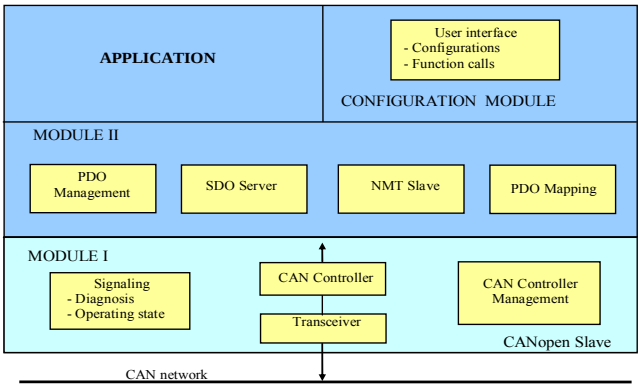

Figure 4. The functional structure of a CANopen slave.

Module I is specific for the numerical system (phyCORE-mpc555, eZdsp-F2808, dsPIC-30F4011) and module II is common of all three systems.

To implement the CANopen protocol it was used both the graphical programming and the classic (textual) programming.

A. Module I implementation on the eZdsp-F2808 or dsPIC-30F4011 numerical systems

The Simulink model visible structure of the slave CANopen communication node is presented in Fig. 5.

The CANOpen Message Receive (dsPIC30F4011 or eZdsp 2808) sub-system realizes the messages reception into the CANopen stack buffer. The messages are transmitted by the CANOpen Message Send (dsPIC30F4011 or eZdsp 2808) sub-system.

They are part of the Module I from the Fig. 4. In the same module there is also the CANOpen Err & Run LED's sub-system which commands the two LEDs of the numerical system.

The stack initialization and its periodical interrogation are realized by the Init CANOpen, and SW_TimerISR sub-systems.

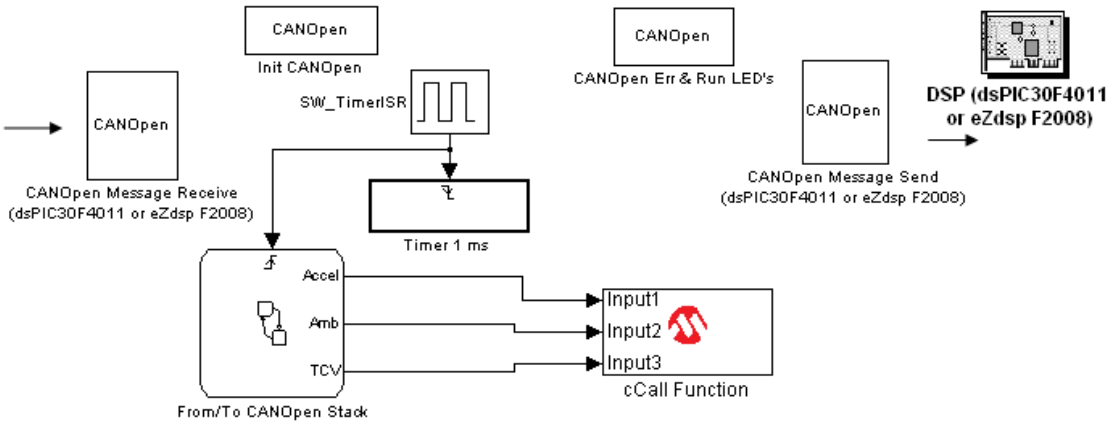

Figure 5. The Simulink model assigned to the slave CANopen communication node.

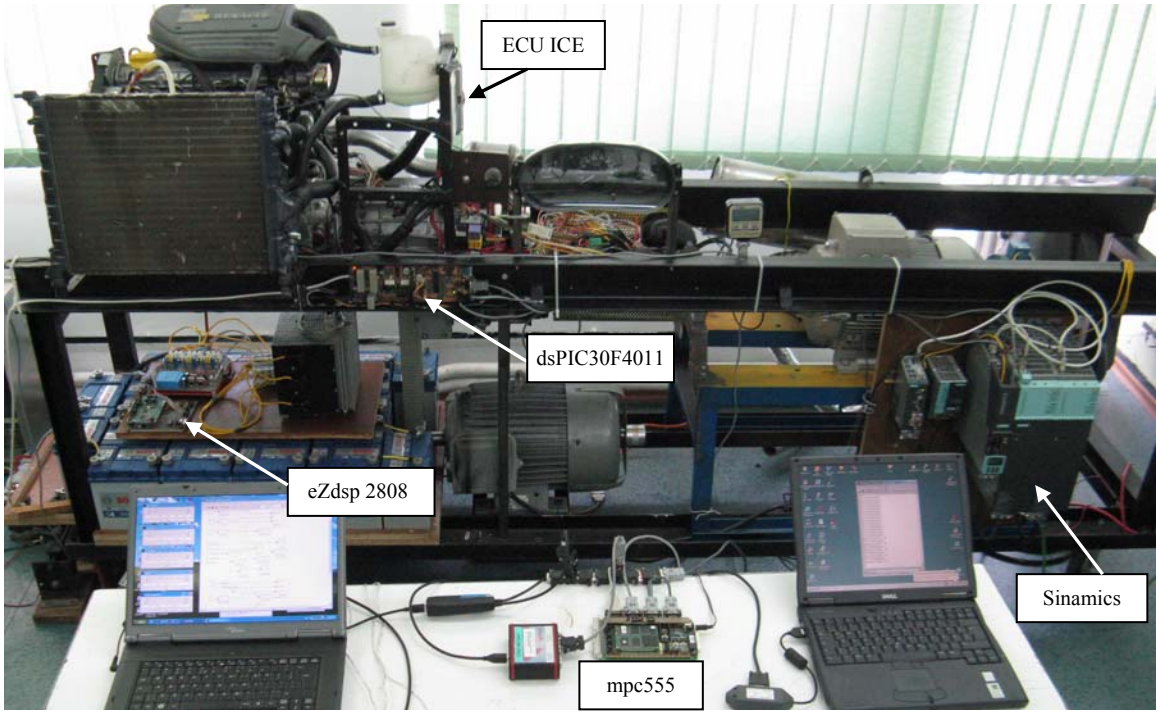

Figure 6. Hybrid electric vehicle experimental model.

The data transfer between the graphical and textual modules is made with global variables which are defined by the state flow chart.

It is to mention that was necessary to interfere with the C-code generating files (Target Language Compiler – TLC) to obtain the necessary functionality.

An important aspect of the CANopen implementation is the generation of relative references of time to administrate the data transfer messages (timestamp) and the administrative data (node guarding, heartbeat).

For this it was used a software which call both the CANopen stack and the timer with 1 ms period.

B. Module I implementation on phyCORE-mpc555 numerical system

The Simulink model for the CANopen node of the second numerical system is similar with the model from Fig. 5, but eZdsp 2808 is changed with phyCORE-mpc 555. Thus, for a user which knows a model it is easy to operate with the other. The communication speed is established with the MPC555 Resource Configuration module.

C. Module II implementation of slave CANopen communication node

The graphical programming is operative and suggestive. It also has limits especially for the complex algorithms processing. In this case the programmer makes a compromise: hardware resources are realized with the graphical libraries and the complex algorithms processing are implemented with textual code lines. The Matlab/Simulink embraces such a combined programming.

Thus, the second module was implemented by a textual programming. The function call is realized with a 1KHz frequency by the SW_TimerISR sub-system. SDO services are assured by the object dictionary SDOResponseTable and by the functions Search_OD (WORD index, BYTE

subindex), Send_SDO_Abort (DWORD ErrorCode) and Handle_SDO_Request (BYTE*pData). The functions Prepare_TPDOs (void) and TransmitPDO (BYTE PDONr) realize the administration of the data transmission messages between the numerical systems

The node initialization is realized by the function CANOpen_Init (BYTE Node_ID, WORD Heartbeat) and the communication network administration (NMT slave) are incorporate into the function CANOpen_ProcessStack(void).

The connections (mapping) between the data on the CAN communication bus can be static realized by the initialization function CANOpen_InitRPDO (BYTE PDO_NR, WORD CAN_ID, BYTE len, BYTE *dat), CANOpen_InitTPDO (BYTE PDO_NR, WORD CAN_ID, WORD event_time, WORD inhibit_time, BYTE len, BYTE *pDat).

The parameterization of a slave CANopen node is realized with a configuration file and thus the node identifier allocation, the signaling time and the PDO number can be immediately established in accordance with the communication network configuration.

The source file of the Module II includes table with SDO responses for the read requests to OD, local and public functions, check parameters. Thus, the communication node assures the functionality of the CANopen protocol.

V. EXPERIMENTAL RESULTS

In fig. 6 is presented the hybrid electric vehicle model realized into the Energy Conversion and Motion Control laboratory of the Electrical and Power Engineering Faculty from Iasi.

Finally several diagrams are presented appropriate electric traction motor and the mechanical load emulator. It was considered a standard operating cycle UDDS (Urban Dynamometer Driving Schedule). A velocity diagram UDDS cycle operation is shown in Fig 7. It is the speed reference for electric traction motor. In Fig. 8 and the fig 9 are presented measured speed and measured active current from electrical motor traction. Also mechanical load emulator is an electrical motor with controlled torque and the torque reference shown in Fig 10. In Fig. 11 is presented the estimated torque from mechanical load emulator.

VI. CONCLUSIONS

For the complex system of the hybrid electric vehicle a hierarchical structure real time control can be used.

All elements of this structure are integrated in a high speed CAN communication network to assure the distributed control. For the hybrid electric vehicle experimental model is used a CANopen network with one master node and four slave nodes witch permits the control of the sub-systems in safe conditions and with improved dynamic performances.

Due to the modularity and scalar structure of the software design the graphical and textual algorithms can be easy reconfigured in order to meet any other requirements of CANopen distributed control [13].

VII. ACKNOWLEDGEMENTS

This work was supported by CNCSIS-UEFISCSU, project no PNII-IDEI cod 622, contract no. 84/01.10.2007.

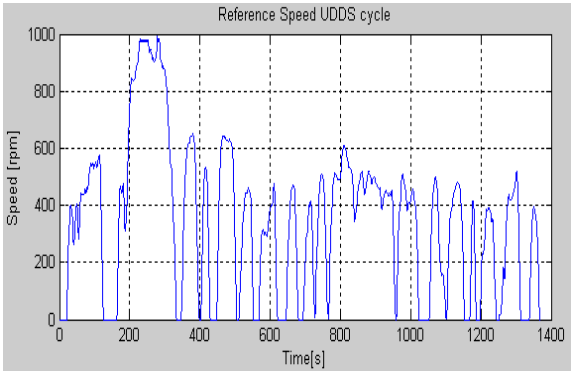

Figure 7. Reference speed for UDSS cycle.

Figure 8. Measured speed for electrical motor traction.

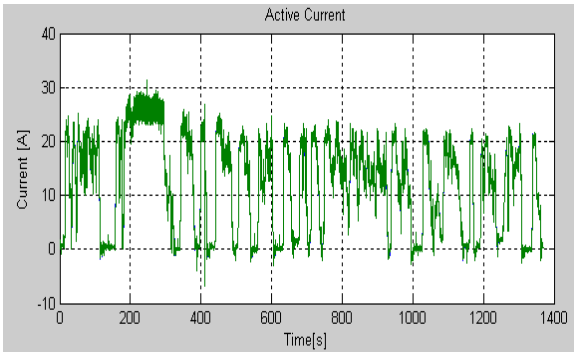

Figure 9. The active current from electric motor traction.

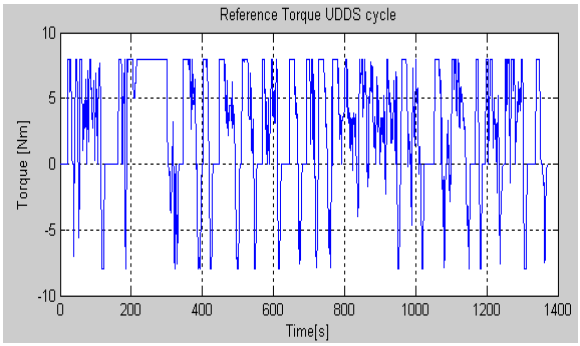

Figure 10. The reference torque for sinamics system.

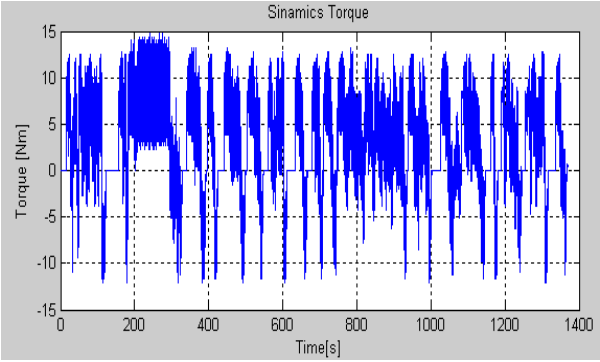

Figure 11. The estimated torque from emulator mechanical load.

REFERENCES

[1] H. Boterenbrood, "CANopen high-level protocol for CAN-bus", NIKHEF, Amsterdam March 20, 2300, Versiunea 3.0.

[2] S. Comigan, "Introduction to the Controller Area Network (CAN)", Texas Instruments Application Report, SLOA101-August 2002, pp. 1-16.

[3] V. R. Chacko, V.Z. Lahaparampil, V. Chandrasekar, "CAN based distributed real time controller implementation for hybrid electric vehicle", 0-7803-9280-9-05/2005IEEE, 247- 251.

[4] J. Duan, J. Xiao, M. Zhang, "Framework of CANopen protocol for a hybrid electric vehicle", Proceedings of the IEEE Intelligent Vehicles Symposium Instanbul, Turkey, June 13-15, 2007.

[5] Gh. Livint, R. Gaiginschi, V. Horga, R. Drosescu, G. Chiriac, M. Albu, M. Ratoi, I. Damian, M. Petrescu, "Vehicule electrice hibride", Casa de Editura Venus, Iasi, 2006.

[6] Siemens, Sinamics, S120 Control Unit and additional system components, 03, 2007 Edition.

[7] CANopen User Manual, Software Manual, July 2004, PHYTE Technology Holding Company

[8] Gh.Livinț, V. Horga, M. Rățoi, "Distributed control system for a hybrid electric vehicle implemented with CANopen protocol" – Part I, Bulletin of the Polytechnic Institute of Iasi, tom LIV (LVIII), FASC. 4, 2008, ISSN 1223-8139, pp. 1019-1026

[9] Gh.Livinț, V. Horga, M. Rățoi , M. Albu, M. Petrescu, G. Chiriac, "Distributed control system for a hybrid electric vehicle implemented with CANopen protocol" – Part II, Bulletin of the Polytechnic Institute of Iasi, tome LIV (LVIII), FASC. 4, 2008, ISSN 1223-8139, pp.1027-1032

[10] Gh.Livinț, V. Horga, M. Rățoi, I. Damian, M. Albu M.,G. Chiriac, "Advanced real rime control algorithms for hybrid electric vehicles optimization", CEEX Program, Simpozion "Contributii Stiintifice", UCP AMTRANS, noiembrie 2008, pp. 209-214

[11] Gh. Livinț, V. Horga, M. Rățoi, M. Albu, G. Chiriac "Implementing the CANopen protocol for distributed control for a hybrid electric vehicle", Proceedings The 8th International Symposium on Advanced Electromechanical Motion Systems, Lille, July 1-3, 2009 , CD., ISBN: 978-2-915913-25-5/EAN: 978291591325

[12] M. Ehsani, Y. Gao, S.E. Gay, A. Emadi, "Modern Electric, Hybrid Electric, and Fuel Cell Vehicles", CRC PRESS, 2005, ISBN 0-8493-3154-4

[13] V.Horga, I.Doroftei, M. Rățoi, "CANopen protocol for the distributed control of an omnidirectional mobile robot", European Journal of Mechanical and Environmental Engineering, Volume 2010, Issue 1.