Paper No. T4-2.5, pp. 1-5

15th INTERNATIONAL SYMPOSIUM on POWER ELECTRONICS - Ee 2009
XV Međunarodni simpozijum Energetska elektronika – Ee 2009
NOVI SAD, REPUBLIC OF SERBIA, October 28th - 30th, 2009

APPLICATION OF MODIFIED TRUETIME SIMULATOR IN CONTROL OF WIRELESSHART NETWORKS

Miroslav Kostadinović, Mile Stojčev*, Zlatko Bundalo, Dušanka Bundalo*****
University of East Sarajevo, Faculty of Traffic and Transportation, Doboj, Bosnia and Herzegovina
*University of Niš, Faculty of Electronic Engineering, Niš, Serbia
**University of Banja Luka, Faculty of Electrical Engineering, Banja Luka, Bosnia and Herzegovina
***UniCredit Bank, Banja Luka, Bosnia and Herzegovina

Abstract: *The usage of modified TrueTime simulator based on the Matlab/Simulink surrounding which simulates the control unit in the execution of tasks of control in wireless networks that work in the real-time has been described in the paper. The TrueTime simulator has been developed on Simulinku and simulates the control system in the meaning of performances, stability and endurance. In the paper are described new possibilities that are implemented in the simulator to improve behavior and usability of TrueTime Wireless Network block such that in the modification was added possibility to select WirelessHart protocol simulation. In order to analyze the behavior of this protocol in the industrial plant, in the paper have been presented simulation results of networked control system in what sensor sends readings to gateway which is responsible for communication with the controller and for sanding of control signal to actuator using WirelessHart network.*

Key Words: ***WirelessHART, TrueTime simulator, Process industry, Control system***

1. INTRODUCTION

The new utility program that will be described in this paper has been implemented to improve behavior and usability of Wireless Network block in TrueTime simulator. Originally have been implemented two types of communication protocols, 802.11b/g (WLAN) and 802.15.4 (ZigBee), and the usage possibility of WirelessHart has been later added. Aim of WirelessHART development is to establish standard for wireless communication for usage in process automatics [1]. WirelessHART makes possible cheap and relatively slow, in the comparison with 802.11b/g, wireless connection with HART-enabled devices. The WirelessHART network protocol is time divided into slots where every slot lasts 10ms. Slots can be dedicated to one node and use TDMA technique for access to the medium or can be divided between a few nodes and use CSMA/CA technique for access to the medium.

TrueTime consists of the library of blocks as shown in the Fig. 1. with following functions:

1. *TrueTime Kernel*: Executes user defined tasks and interrupts which for example present input/output tasks, control algorithms and network interfaces.
2. *TrueTime Network*: This block is used to simulate access to the medium and transfer of packages according to the chosen network model.
3. *TrueTime Wireless Network*: Function of this block is similar as of the TrueTime Network block, but instead of wire network it uses wireless network.
4. *TrueTime Battery*: This block is used to simulate the battery-powered supply.
5. *ttGetMsg*: It is used for reception of messages from the network.
6. *ttSendMsg*: It is used for sending messages on the network.

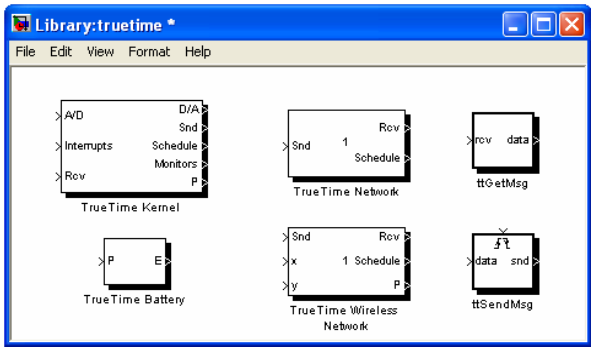

Fig. 1. *TrueTime library of blocks*

2. ADJUSTMENT OF KERNEL AND WIRELESS NETWORK BLOCK

S-function of the kernel block simulates the CPU with simple but flexible real-time kernel, A/D and D/A converters, network interfaces and canals of external interrupts. Realization of tasks and interrupts is defined by user written function codes which can be written in C ++ or like the m-files. Control algorithms can be defined graphically using usual Simulink block diagrams [2].

Here has been described and illustrated adjustment of Simulink blocks and TrueTime kernel. On the very beginning from the TrueTime library of blocks it is

transferred the TrueTime kernel block into empty window of new created model, like in Fig. 2. Functions of input and output ports on the kernel block are following:

1. A/D port is used for conversion of analog signal in the digital form. If there is more than one analog input it is used multiplexer.
2. Interrupts input port is used for reception of interrupts from external resources, like are the switches.
3. Rcv input port is used for messages reception from the network.
4. D/A output port is used for conversion of digital in analog signal. Similarly as for A/D inputs, if it is needed output for more than one signal then it is used demultiplexer.
5. Snd output port is used for sending messages through the network.
6. Schedule output port is used for graphic presentation of executions of planned tasks. It can present the status of different tasks that can be following: running (high), ready (middle) or idle (low).
7. Monitor output port is used for determination of status of different semaphores.
8. P output port is used for simulation of devices which are supplied by battery.

Fig. 2. *TrueTime Kernel*

Placing the mouse pointer on the TrueTime Kernel block and after double click it is opened new window „Function block parameters“ which is used for adjusting of kernel block parameters.

Configuration of the TrueTime Kernel has been presented in the Fig. 3., where the input ports „Interrupts“ and „Rcv“ are connected with grounding which is in „Sources“ menu of Simulink library. Output ports „Snd, „Monitors” and „P“ are connected with terminators which are in „Sinks“ menu of Simulink library. Output port „Schedule“ has been connected with the oscilloscope which has been marked with „Schedule“ and is in „Sinks“ menu of Simulink library.

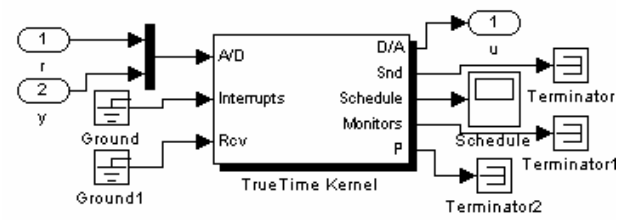

Fig. 3. *Configuration of TrueTime Kernel ports*

A/D input port has been connected with two input ports marked with r and y, via the multiplexer which is in „Signal Routing“ menu of Simulink library, and D/A output port of kernel has been connected on output port and marked with „u“.

Wireless Network block makes possible simulation of communication between two nodes, and in the original TrueTime simulator have been supported two communication protocols: IEEE 802.11b/g (WLAN) and IEEE 802.15.4 (ZigBee), while the WirelessHART protocol has been subsequently added [3], [4].

Model Radio transmission includes the support for:

1. Ad-hoc wireless networks.
2. Isotropic antenna.
3. Impossibility of simultaneous sending and receiving of messages.
4. Modeling of path of loss of radio signal like $1/d^a$ where d is distance in meters and a is parameter that has been chosen for different environments.
5. Interference from other communication appliances.

In the modification of the TrueTime simulator it has been added possibility of selection of simulation of WirelesHart protocol in the framework of Wireless Network block such that are used parameters which are uses by Wlan and ZigBee, but with the supplement of following three fields:

1. field for setting up of value of number of transfer canals,
2. size of slot,
3. length of superframe.

3. SIMULATION MODEL OF DC SERVO MOTOR CONTROL

Model of the control system consists of three devices (nodes), like in the Fig. 4., which have been placed on mutual distance of 20 meters and every of them has been presented with TrueTime kernel and have been connected by one Wireless Network block.

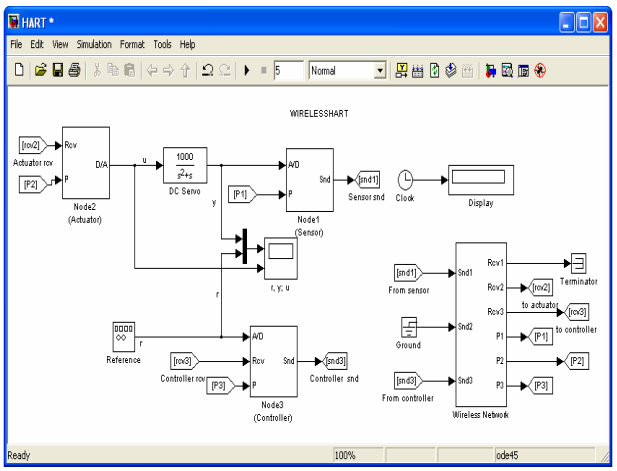

Fig 4. *Model of control system*

In this control system exists only one loop in which sensor periodically converts analog signal from the process into the digital value and sends it to the controller. Controller, after receiving the message from the sensor, calculates the output according to the suitable control algorithm and sends control signal to the actuator using WirelessHart network, like in Fig. 5. Actuator converts the received control signal in analog one and is moved.

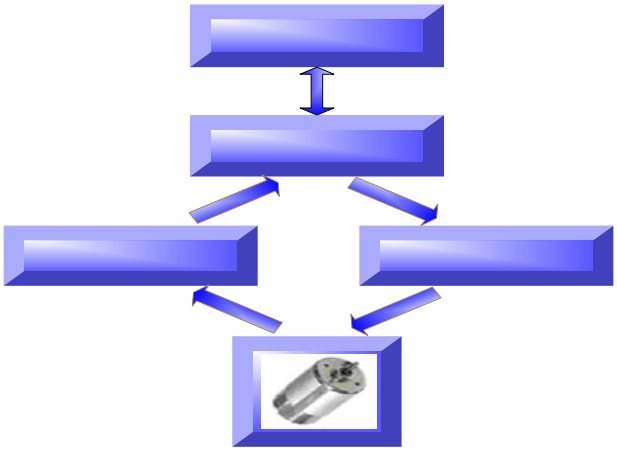

Fig. 5. *Scheme of communication between devices*

In order to be able to use WirelessHART network, transfer function of system must have the time constant at least of order of tenths of milliseconds. This is imposed by the MAC protocol which uses TDMA technique with time intervals of 10 ms. In the every interval it is possible only one transfer (in the framework of the same canal) so that, when we speak about the control system, the smallest delay between the reading and activation is 20 ms. It is possible if the task of actuator executes immediately after the task of sensor. In the first period sensor (S) sends readings to the controller (C) and in the second period controller sends control signal to the actuator (A). If the run time of task of controller is equal to the total time period (10 ms), it is the best to leave one free space between the reading and the reaction in order to enable the controller to finish its calculations.

For the adjusting of simulation of WirelessHART protocol it has been chosen the shortest possible superframe. In fact the WirelessHART has the minimal length of superframe of 0.5s what means 50 time slots by 10ms each. It is obvious that for control of system it is necessary to „listen“ it more often than 0.5 s.

Superframe has been created in a way that has been presented in Fig. 6., so that the time slot between the task of sensor and actuator is left empty. This has been done from the reason to enable controller to execute calculations, and the sensor sends the measured values every 0.03 s.

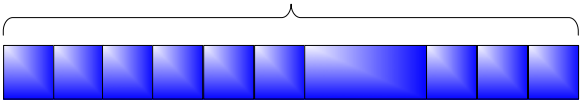

Fig. 6. *WirelessHART superframe*

4. ADJUSTMENT OF SIMULATION ENVIRONMENT

In the TrueTime simulation model which has been presented in the Fig. 4. has been designed the networked control system where have been connected three network devices using communication via WirelessHART network [2], [4], [5]. Double-click on Wireless Network block (Fig. 7.) opens the dialog window in which we put the number of devices/knots (three) and select the WirelessHart protocol.

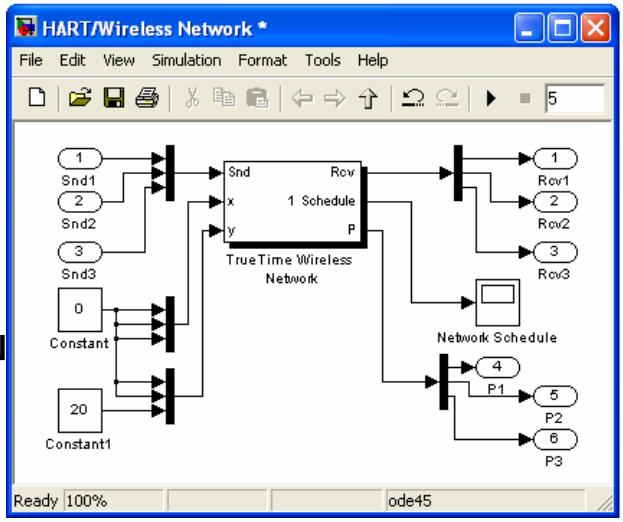

Fig. 7. *TrueTime Wireless Network block*

4.1. Initialization of sensor

Network devices/nodes have been simulated in the subsystem with the TrueTime kernel block. Details of subsystem for sensor/node1 have been given in the Fig.8. Sensor/node1 uses one A/D converter in the input part and one network output (Snd) at the output.

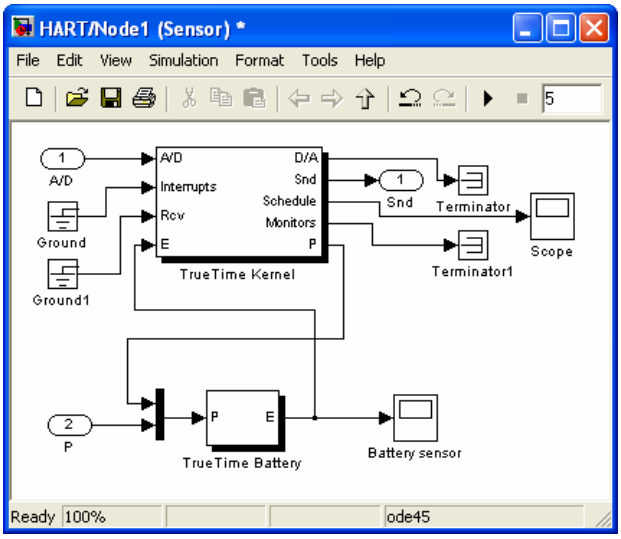

Fig. 8. *TrueTime kernel block of sensor*

Sensor/node1 periodically converts information from the analog input and sends information to the network, using the function sensor_init which has been called in the TrueTime kernel, and has been presented in the Fig.9.

The every device/node which contain the TrueTime kernel block must be started by the function ttInitNetwork (the identification number of device, name of function of network interrupt). It sets the network address of device/node and also defines which interrupt will be performed when the new message arrives. If it is necessary that periodically be perform the task on the device/node, it is used the function ttCreatePeriodicTask (time period, priority, name of m- file, local information for the device).

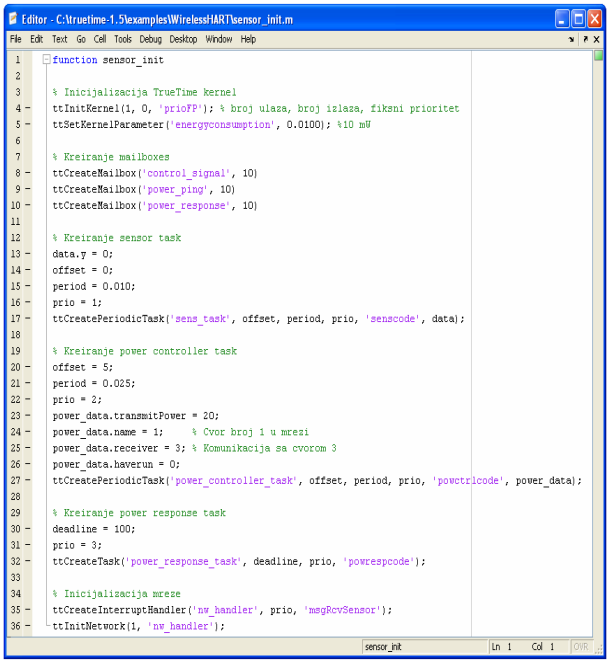

Fig. 9. *Function sensor_init*

4.2. Initialization of actuator

Details of subsystem for actuator/node2 have been given in Fig. 10. Actuator/node2 uses one network input (Rcv) in the input part and one D/A converter on the output.

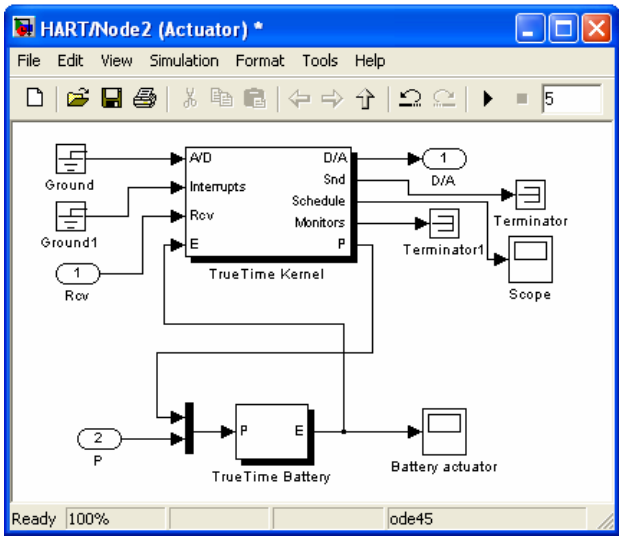

Fig. 10. *TrueTime kernel block for actuator*

The task and interrupts have been created in the function for initialization actuator_init. The function actuator_init, shown in Fig. 11., does not contain declaration of periodical task, i.e. does not use the function ttCreatePeriodicTask, because it does not send information to the network but only accepts information from the network. If the device would accept the information from the network then it would be needed to determine function of interrupts ttCreateInterruptHandler (the unique name, priority, name of executed m-file), in order to control accepted messages from the network.

Actuator is „connected“ on the network using function ttInitNetwork, offering identification number of device and name of interrupt (rcv_hdl) which is needed to be performed when the message arrives on the device.

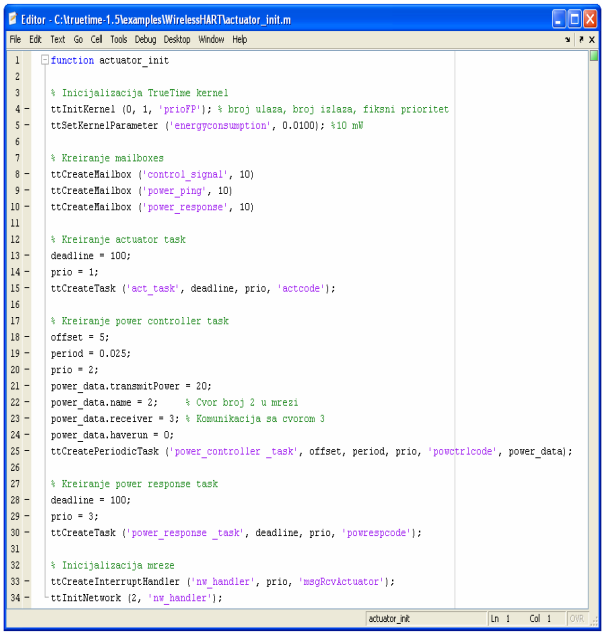

Fig. 11. *Function actuator_init*

In the ttInitKernel function kernel is started specifying number of A/D and D/A converters, policy of scheduling and time of simulation. Built-in function of priorities prioFP determines fixed schedule of priorities.

4.3. Initialization of controller

Controller receives messages from the sensor and sends data back on the network i.e. to the actuator. Details of subsystem for the controller/node have been given in Fig. 12., where is used one A/D converter and one network input (Rcv) on the input part and one network output (Snd) on the output part.

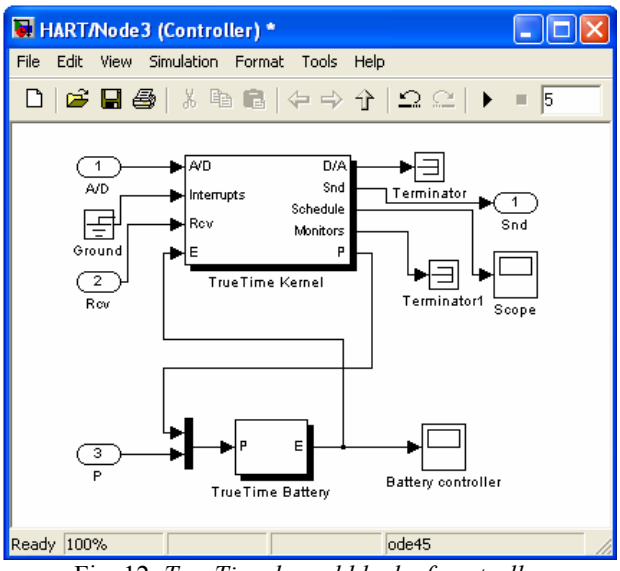

Fig. 12. *TrueTime kernel block of controller*

After adjusting parameters of kernel block it is necessary to create m-file ctrlcode (Fig. 13.), which calculates the control output signal and uses equations for the PD controllers.

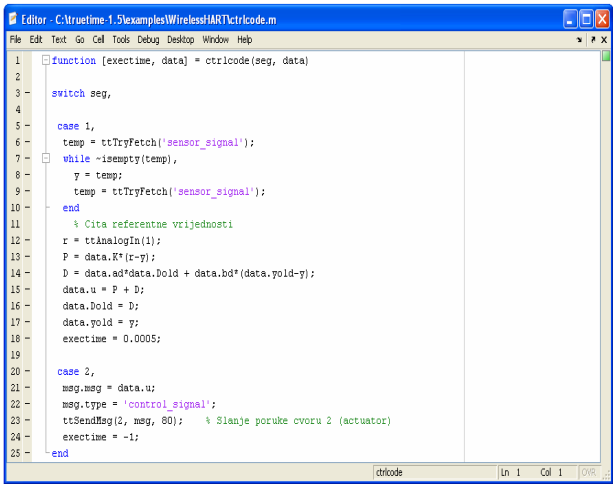

Fig. 13. *Function ctrlcode*

After creation of the previous m-files it is necessary to create and record the controller_init m-file.

When are connected all previously processed Simulink blocks it is obtained the model of control system (like in Fig. 4.), whose the wireless communication is realized by the WirelesHART protocol [3], [4].

5. SIMULATION RESULTS

Simulation has been performed in the Simulink surrounding using the modified TrueTime simulator and in order to use WirelessHART network protocol it is necessary to check if the modeled system has the time constant of order of tenths of milliseconds. In the case that the system is faster then this network protocol could not be used.

If the simulator does not report errors on the occasion of starting simulation, by double-click in the model (Fig. 4.) on the oscilloscope opens the window like in Fig. 14. The new opened window makes possible to follow result of simulation of data transfer using WirelessHART protocol implemented on the model of control system for DC servo motor control.

Fig. 14. *Simulation result*

Realization of schedule of execution of tasks in the WirelessHART network has been shown in Fig 15. and can be in details analyzed where high level of signal means the sending, middle level of signal presents the waiting and low level of signal means idle.

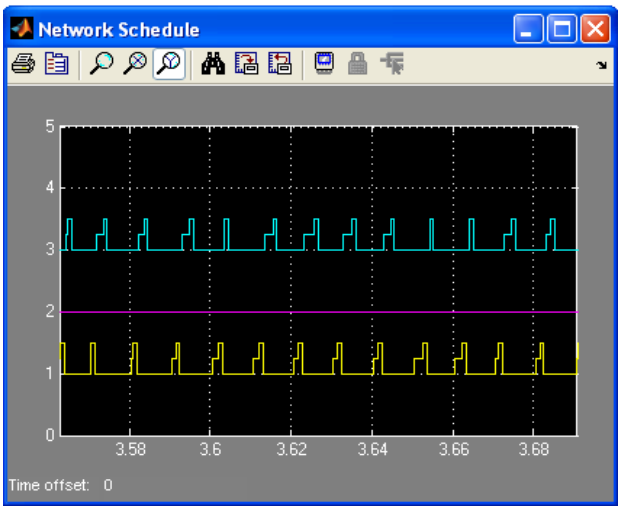

Fig. 15. *Scheduling of execution of tasks in the network*

6. CONCLUSION

In this paper has been described the new way of implementation of WirelessHart with the modified TrueTime simulator based on the MATLAB/Simulink, which can simulate the control device in the execution of tasks in real-time systems, networks and dynamic plants. In details have been explained all phases of adjusting TrueTime simulation surrounding necessary for the analysis of behavior of WirelessHART protocol implemented on the example of control system with three nodes for control of DC servo motor. Also, in the form of graphs have been illustrated results of simulations for control and scheduling of executions of tasks in WirelessHart network.

7. REFERENCES

[1] HART Communication Foundation: "*Control with WirelessHART*", HCF LIT-127, Revision 1.0, June30, 2008.

[2] A. Cervin, D. Henriksson, M. Ohlin: "*TrueTime 2.0 beta-Reference Manual*", Department of Automatic Control, Lund University, Lund, Sweden, January 2009.

[3] M. Kostadinović, M. Stojčev, Z. Bundalo, D. Bundalo: "*Control of WirelessHART network*", Internatonal Symposium INFOTEH-JAHORINA 2009., Jahorina, 2009. (in Serbian)

[4] M. Kostadinović, M. Stojčev, Z. Bundalo, D. Bundalo: "*Design, Implementation and Simulation of WirelessHart Network*", 9th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services TELSIKS 2009., Niš, 2009.

[5] M. Andersson, D. Henriksson, A. Cervin and K. E. Årzén: "*Simulation of wireless networked control systems*", in Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference ECC 2005. Seville, Spain, December 12-15 2005.