Integrating wireless systems into process industry.

Conference Paper · January 2011

Source: DBLP

M. Kostadinovic
University of East Sarajevo

Integrating Wireless Systems Into Process Industry

Miroslav Kostadinovic*, Zlatko Bundalo**, Dusanka Bundalo***, Ferid Softic**
*Faculty of Traffic Engineering/University of East Sarajevo, Vojvode Mišića 52., Doboj, Bosnia and Herzegovina
**Faculty of Electrical Engineering/University of Banja Luka, Patre 5, 78000 Banja Luka, Bosnia and Herzegovina
***Faculty of Philosophy/University of Banja Luka, Vojvode Bojovica bb, 78000 Banja Luka, Bosnia and Herzegovina
kostadinovicm@gmail.com

Abstract - Now there is a trend of replacement of wired communication systems in process industry with the new developed wireless communication systems due to their more efficient and cheaper ways of connection and use. The conducted research and obtained results in the field of integration of wireless systems in process industry are described in the paper. The really existing known implementations of industrial communication networks and a critical review of their technical capabilities are shortlly considered first. Then, the requirements for the implementation of a modern management system in the processing industry is specified and the models of integration HART and WirelessHART networks are proposed. The assessment of managerial performances of the integrated network model using the results of simulation for the implementation of network and software tools for configuring, planning and management of wireless networks is also given.

I. INTRODUCTION

WirelessHART (***H****ighway* ***A****ddressable* ***R****emote* ***T****ransducer*) protocol in the process industry is supported by already installed HART devices and is a future choice for the automation of industrial processes [1]. The reasons that justify their integration are as follows:

- Support a large number of HART devices that can operate within WirelessHART network.
- Installed HART networks are able to communicate with WirelessHART networks.
- New WirelessHART devices are compatible with both protocols.
- In the process industry are used the HART devices that support the new WirelessHART protocol.

The needs for integration justifies also HCF (***HART*** ***C****ommunication* ***F****oundation*) which provides direct and indirect way for the integration of HART and WirelessHART protocol recommending the following three ways:

- Installed HART networks are able to communicate with WirelessHART networks.
- Integration of HART network with WirelessHART network.
- Integration of WirelessHART devices with HART network.

II. SAFETY IN INTEGRATED NETWORKS

One of the ways for connecting HART devices with WirelessHART networks is the use of conventional adapters/ However, by expanding capability of the adapters they can be used as a integrator, i.e. tool for security integration of HART and WirelessHART network. Gateway can also be used as an integrator. But if the safety and reliability are the main goals of integration the use of special adapter as an integrator is a better choice because of the following reasons:

- The adapter is designed to allow connection of multiple current loops, i.e. in multidrop architecture more than one device can be connected into the current loop and then to one adapter, but in the point to point architecture can be connect only one device to the current loop.
- In contrast to the gateway, adapters are portable and change in physical location does not require additional cabling.
- Adapters can be connected to any HART components, but security will be greater if the adapter is connected directly to one of the master device that is connected with the actual device.
- If the gateway is used for integration, its architecture is more complex and more expensive. Such gateways are specialized and are used only where is required integration.

All these reasons justify the usage of adapters as a network integrators which can be used for secure integration of HART and WirelessHART networks.

III. DESIGN AND IMPLEMENTATION OF MODELS

Here is described how to use the original TrueTime simulator based on Matlab/Simulink, which simulates the controller in performing tasks in real-time systems, networks (wired or wireless networks) and dynamic drives.

TrueTime is not constituted only from library with blocks that are shown in Fig. 1, but also from collections of C + + functions with the corresponding MATLAB

MEX interface. Some functions allow the use of simulation to create tasks, interruptions, control, timers etc.. Other functions are real-time referred in the code during the execution of the task and allow AD / DA conversion, sending and receiving of messages, etc.

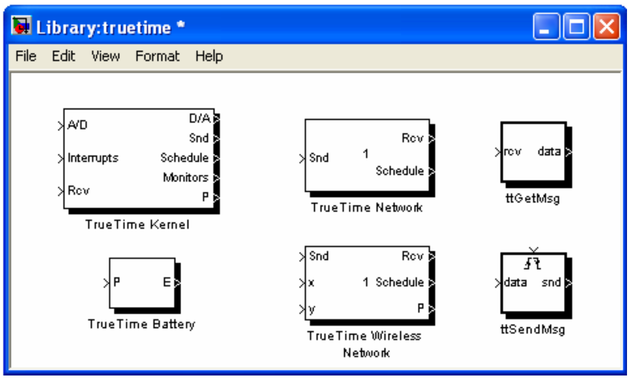


Fig. 1. TrueTime library of Simulink blocks

TrueTime was developed in Simulink, which takes care of the control system in terms of performance, stability and endurance and is primarily intended for use with MATLAB / Simulink [2] and contains a library of blocks as shown in Fig. 1 that have the following functions:

- TrueTime Kernel: Performs user-defined tasks and interruptions, for example. represent the I/O tasks, control algorithms and network interfaces.
- TrueTime Network: This block is used to simulate the access to the medium and packet transfer according to the selected wire network model.
- TrueTime Wireless Network: The function of this block is similar to the TrueTime Network block, so that instead of wire network uses wireless network.
- TrueTime Battery: This block is used to simulate battery power supply.
- ttGetMsg: It is used for receiving messages from the network.
- ttSendMsg: It is used for sending messages from the network.

Model of the control system has three devices, i.e. nodes, such that in the selection of WirelessHART protocol is unified function of controller and gateway. Each of these devices is presented with TrueTime kernel, and are at a distance of 20 meters and are connected by True Time Network and Wireless Network block [2] as in Fig. 2.

In this model of the control system there is a loop in which the sensor is periodically converting analog signal from the process in the digital value and sends it to the controller. Controller, after receiving a message from the sensor, calculates the output in accordance with appropriate control algorithm and sends a control signal to the actuator using the HART and WirelessHART network, as in Fig. 2. The actuator converts the received control signal into an analogue one and was running [3,4,5].

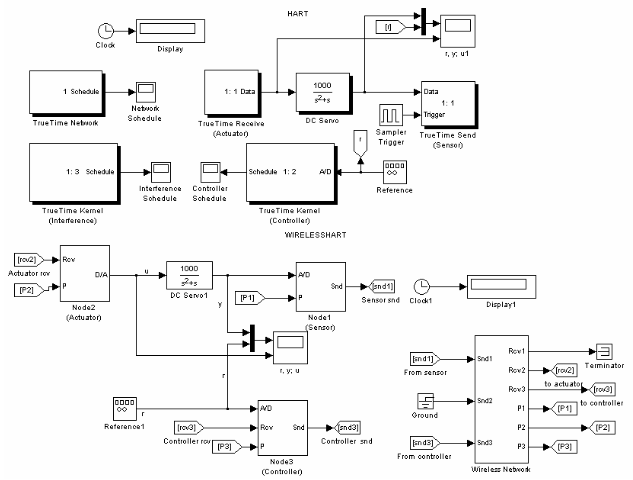


Fig. 2. Model of control system

A. *User Interface Architecture*

Before simulating operation of the network, it is needed to ensure the user enters into the network through the user interface. Fig. 3. shows a number of communication links between devices realized in one communication round. The meaning of individual blocks is as follows:

- The first processing block allows the user to define the number of super frames and length of each of them in the network. All super frames are of different length.

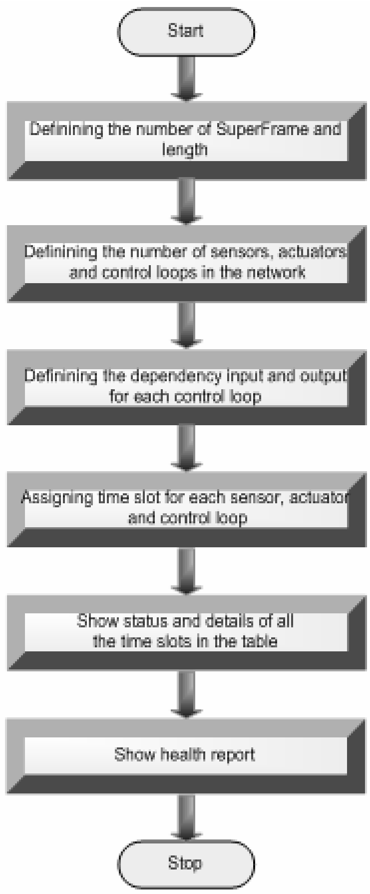


Fig. 3. User interface architecture

- The second block allows the user to define the number of sensors, actuators in the network and the number of control loops in the network.
- The third block allows the definition of dependent input and dependent output for each control loop. Dependent input is data obtained by the sensor device, for which control signals should be calculated.
- The fourth block is distributing to each device in the network a certain time slot in the appropriate super frame.
- The fifth block allows the tabular presentation of the status and details of each time slot of all super frames.
- The sixth block allows presentation of information about the schedule of execution of tasks of individual devices in the network, as well as of errors and warnings.

IV. CONFIGURING NETWORK DEVICES

In TrueTime simulation model which is shown in Fig.2. was designed control systems where are connected three network devices using wired and wireless communication and using HART protocol [6,7].

TrueTime kernel configuration is shown in Fig. 4 where the input ports "Interrupts" and "Rcv" were connected to the earth which is located in the "Sources" menu of Simulink library. Output ports "Snd", "Monitors" and "P" are connected to terminators located in the "Sinks" menu of Simulink library. "Schedule" output port is connected to an oscilloscope that is marked with a "Schedule" and is located in the "Sinks" menu of Simulink library.

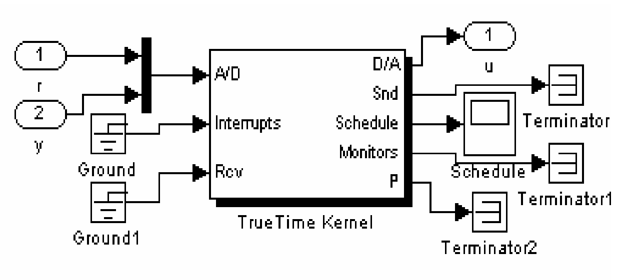


Fig. 4. Configuration of TrueTime kernel ports

A/D input port is connected to two input ports labeled with r and y through a multiplexer located in the „Signal Routing“ menu of Simulink library, and D/A kernel output port is linked to the output port marked with "u".

Double click on TrueTime Network and Wireless Network block in Fig. 2. opens a dialog window in which we enter the number of devices/nodes and choose WirelessHART and HART protocol.

In this model of system exist three different devices: sensor, controller and actuator, which must be initialized prior the use in the network [8,9].

A. *Initializing of sensors*

Network devices/nodes are simulated in the subsystem with TrueTime kernel block. Details of sunsystem for sensor/node1 are given in Fig. 5. Sensor/node1 uses one A/D converter in the input section and one network output (Snd) at the exit. Sensor/node1 periodically converts the information from the analog output and sends the information to the network, using the function sensor_init that is invited in TrueTime kernel and is shown in Fig. 6.

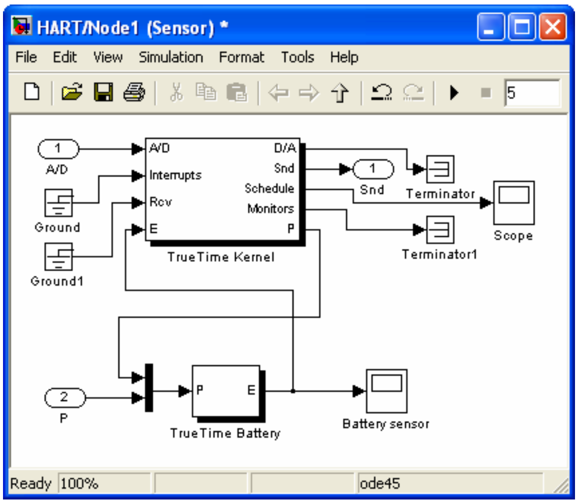
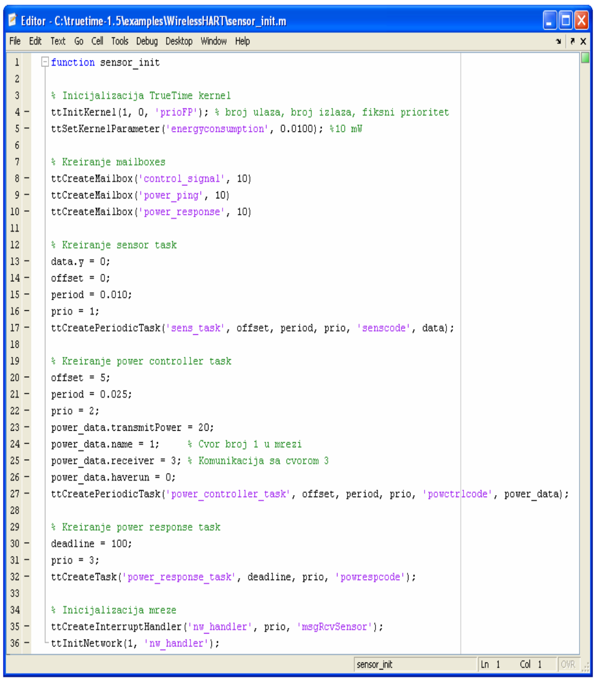


Fig. 5. TrueTime kernel block of sensor

Initialization of devices can be made by writing a piece of code in a separate M-files for each device, so that we have to define in the network the number of analog inputs and outputs for each node and to planne tasks execution [2]. This can be achieved by writing the function 'ttInitKernel', in the following form: ttInitKernel (nbrOfInputs, nbrOfOutputs, scheduling policy). In Fig. 6. we have shown M-file for initialization of the sensor where the previous function is implemented.




```
function sensor_init

% Inicijalizacija TrueTime kernel
ttInitKernel(1, 0, 'prioFP'); % broj ulaza, broj izlaza, fiksni prioritet
ttSetKernelParameter('energyconsumption', 0.0100); %10 mW

% Kreiranje mailboxes
ttCreateMailbox('control_signal', 10)
ttCreateMailbox('power_ping', 10)
ttCreateMailbox('power_response', 10)

% Kreiranje sensor task
data.y = 0;
offset = 0;
period = 0.010;
prio = 1;
ttCreatePeriodicTask('sens_task', offset, period, prio, 'senscode', data);

% Kreiranje power controller task
offset = 5;
period = 0.025;
prio = 2;
power_data.transmitPower = 20;
power_data.name = 1;     % Cvor broj 1 u mrezi
power_data.receiver = 3; % Komunikacija sa cvorom 3
power_data.haverun = 0;
ttCreatePeriodicTask('power_controller_task', offset, period, prio, 'powctrlcode', power_data);

% Kreiranje power response task
deadline = 100;
prio = 3;
ttCreateTask('power_response_task', deadline, prio, 'powrespcode');

% Inicijalizacija mreze
ttCreateInterruptHandler('nw_handler', prio, 'msgRcvSensor');
ttInitNetwork(1, 'nw_handler');
```


Fig. 6. Sensor_init function

By the user is defined the offset of the channels, the destination address, the direction of communication, the connection characteristic, ID of frame for each time slot and creates a table of data containing all information about the hubs. After that are created periodicall tasks that are active every 0.01s, which is achieved writing function 'ttCreatePeriodicTask'. When we call this function we must define the parameters shown in the following form: ttCreatePeriodicTask ('Task Name', offset, period, prio, 'function name', data):

- Task name parameter is requiring to provide name for each task, so there is no possibility of the same names of tasks in the network.
- Offset is the time that is used to postpone or to delay of execution of the task.
- Period is defined in seconds and is used to periodicall excitation.
- Function name parameter is used for giving name to a function that should be called during activating task.
- Data is used when the user wants to pass some parameters for calling of function and then have to be called a function that manages interrupts for the nodes.

To use the interrupt function it must be created a interrupt task, define the priority of the function and the function name that needs to be performed at the interrupt and what can be achieved by writing functions: ttCreateInterruptHandler (handler name, priority, function name). Each device/node that contains the block of TrueTime kernel must be started by the function ttInitNetwork [2]. It sets the network address of the device/node and also defines what interrupt will be performed when new message arrives. If it is necessary to periodically perform a task on the device/node used is the function ttCreatePeriodicTask.

B. *Initializing of actuators*

Details of actuator/node2 subsystem are shown in Fig. 7 by using one network input (Rcv) in the entrance area and one D/A converter at the output.

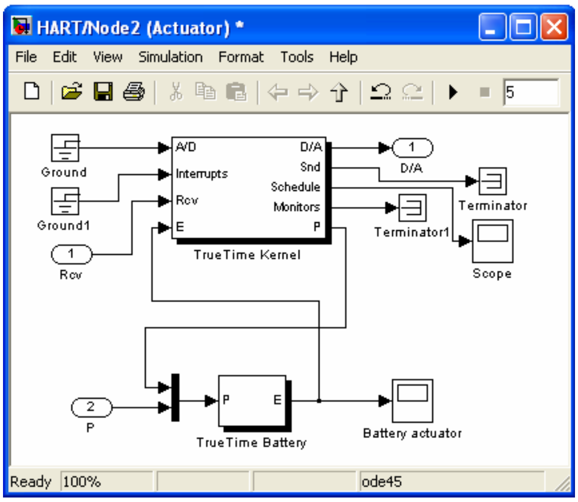


Fig. 7. TrueTime kernel block of actuator

Actuator_init function in Fig.8 does not contain a declaration of a periodic task, i.e. does not use the function ttCreatePeriodicTask, because it does not send information onto the network than just receives it from the network. If the device should receive information from the network it should be only specify the function of handler of interrupt ttCreateInterruptHandler to control received messages from the network. Actuator is "connected"to the network using the function ttInitNetwork providing device identification number and name of interruption handler (rcv_hdl) which should be performed when a message arrives at the device. In ttInitKernel function the kernel is started by specifying the number of A/D and D/A converters, planning policy and simulation time. Built-in priority function prioFP determines fixed priority planning.

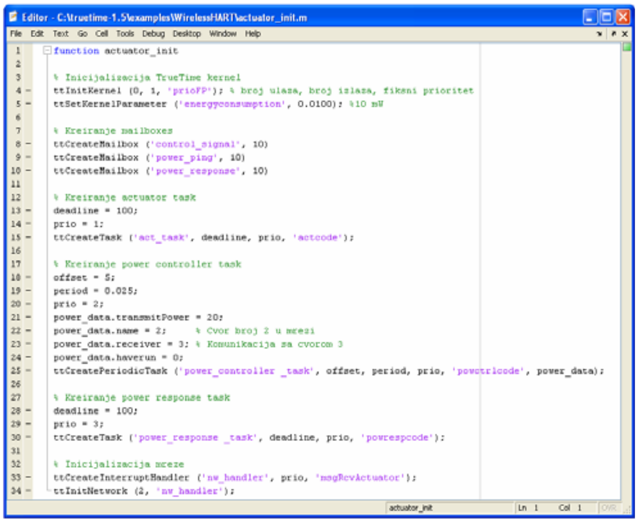


Fig. 8. Initializing of actuator

C. *Initializing of controller*

The controller receives messages from sensors and sends data back to the network, ie. to actuator. Details of controler/node3 subsystem are given in Fig. 9. where are used one A/D converter and one network input (Rcv) in the entrance area and one network output (Snd) in the output section.

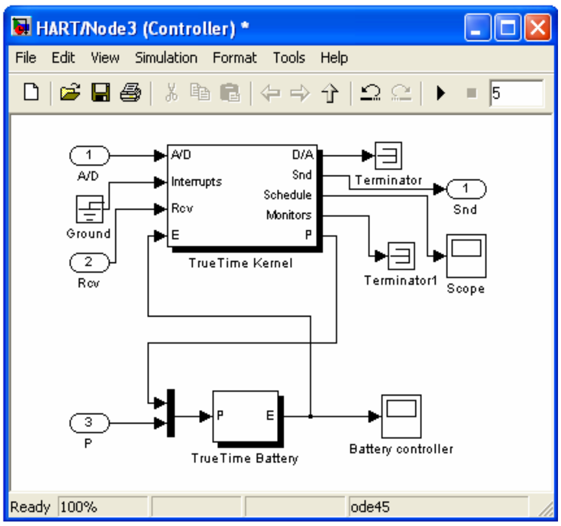


Fig. 9. TrueTime kernel block of controller

After setting the parameters of the kernel block it is necessary to create ctrlcode M-file which is shown in Fig. 10. This written code calculates control output signal by using the equations for the PD controllers [6,7].

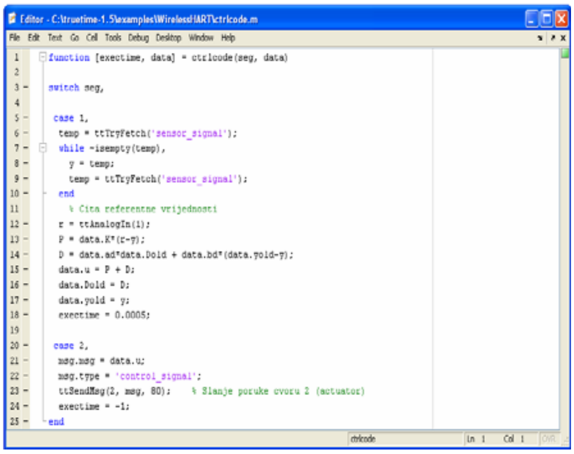


Fig. 10. Function ctrlcode

When the controller receives a message from the network happens the interrupt, ie. is called the m-file function msgRcvCtrl as in Fig. 11 which performs the task using the line of code: ttCreateJob ('pid_task').

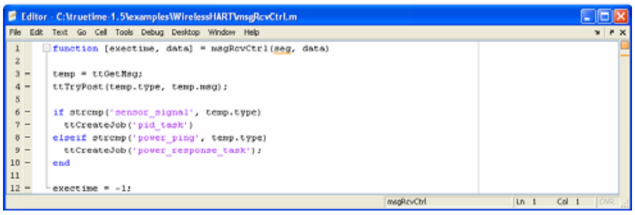


Fig. 11.. Function code for receiving messages of controller

After creating the previous m-files it is needed to create and record controller_init m-file as in Fig. 12 which contains line of code defining the kernel so it has 1 input, 0 outputs and fixed priority for planning: ttInitKernel (1, 0, 'prioFP').

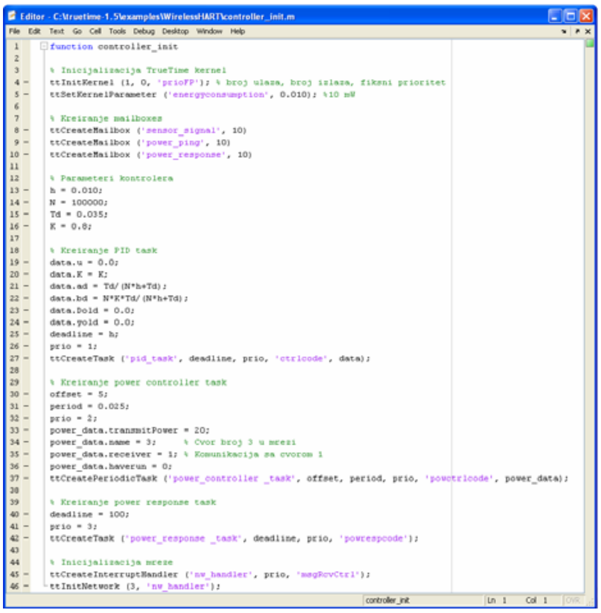


Fig. 12. Initializing of controller

The task represents the main structure in TrueTime simulation environment. In the same m-file are defined the attributes and data of task for PD controller. Based on that can be created task by adding the following line of code: ttCreateTask ('pid task' deadline, prio, 'ctrlcode' , data). Command ttCreateTask creates task 'pid task' that calculates the control output signal using ctrlcode m-file. Since the task is not periodical are used interrupt 'nw handler' created in the command ttCreateInterruptHandler. When are connected all previouselly described Simulink blocks the model of control system is obtained such is shown in Fig. 2 in which the wire and the wireless communications are implemented with HART protocol.

V. SIMULATION RESULTS

To analyze diversely of the behavior of the two (HART and WHART) protocols in the industrial plant it will be presented simulation of networked managing system whose scheme is shown in the Fig.2. All the simulations are performed in Simulink surrounding using the modified TrueTime simulator [1]. If the simulator does not report errors when starting the simulation by double click on the oscilloscope in the model opens the window sa shown in Fig 13 and Fig 14.

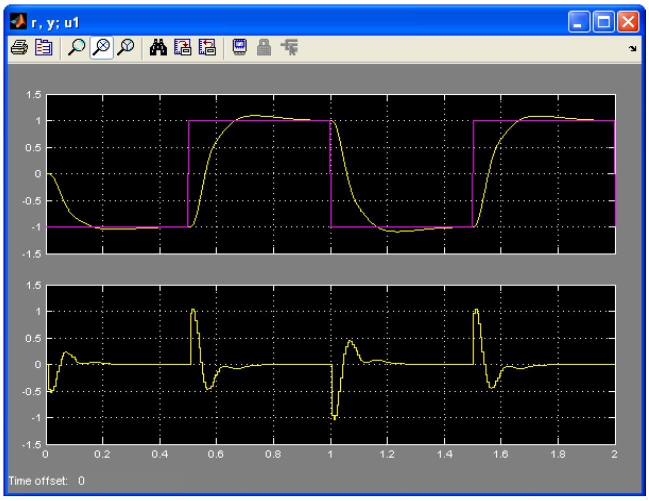


Fig. 13. Simulation results for wired communication

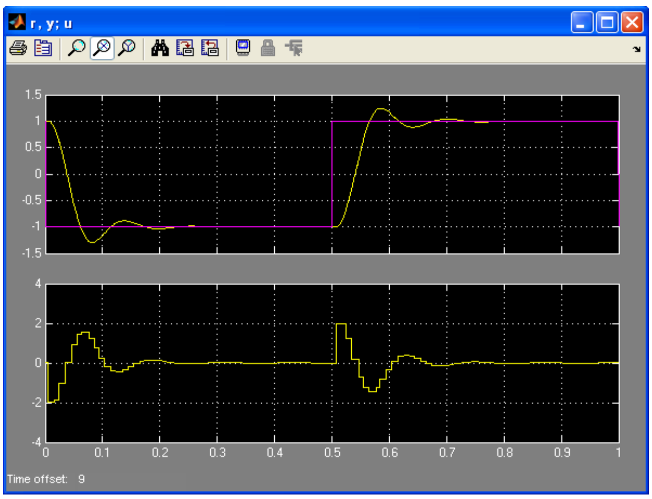


Fig. 14. Simulation results for wireless communication

The new opened window allows to monitor the results of the simulation of data transfer using the HART protocol implemented on the model of control system for wired and wireless communications. In this managing system it is only one loop in which sensor sends readings off to gateway which is responsible for communication with the controller using the protocol for the wire data transfer and for the sending managing signal to the actuator using the wireless network.

During choice of the transfer function of system which WirelessHART can control only are processes which have time constant at least of tenths of milliseconds. This is imposes by the MAC protocol which uses TDMA (***T**ime **D**ivision **M**ultiple **A**ccess*) with time intervals of 10 ms. In every interval it is possible only one transfer (in the frame of the same canal) so that, when it is spoken about the managing system, the smallest delay between reading off and excitation is 20 ms. It is possible if the task of actuators executes immediately after the task of sensors, as in Fig. 15.

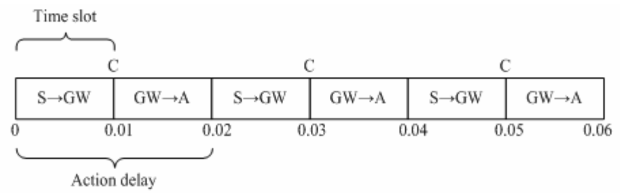


Fig. 15. Illustration of communication in WirelessHART

In the first time period sensor (S) sends readings off to the gateway (GW) and in the second time period gateway sends the managing signal to the actuator (A). Controller (C) executes the command between time periods of duration of tasks of sensor and actuator [1].

From the Fig. 15. it can be noticed that the controller is not needed time space for the execution of task. That is because of the fact that the controller with gateway communicates across the wire. If it is the run time of task of controller equal to the total time period (10 ms), the best is gives one free period between the reading off and reaction in order to enable that the controller finishes its calculations. The output is the square wave with the amplitude 1, time period of 5 seconds and working cycle of 50% as in Fig 14., and the used PID-controller[1].

VI. CONLUSION

In this paper are explained all the phases of setting TrueTime simulation environment necessary for analyze of the behavior of integration of HART protocol implemented in the example of the control system with three nodes. In the paper in the form of graphs are illustrated obtained results of simulation for the management and planning of tasks performing using the HART protocol implemented on the model of management system for wired and wireless communication. It can be concluded that in this paper is described a new method of implementation of integration of HART devices and WirelessHART network using modified TrueTime simulator based on Matlab/Simulink which can simulate the controller in performing tasks in real-time systems, networks and dynamic drives.

REFERENCES

[1] HART Field Communication Protocol, “Application guide”, *HCF* 2006.

[2] Andersson, M., D. Henriksson, A. Cervin “TrueTime 1.5—Reference Manual”, *Department of Automatic Control*, Lund University, Sweden, January 2007.

[3] HART Communication Foundation, “Control with WirelessHART”, *HCF LIT-127*, Revision 1.0, June30, 2008.

[4] HART Communication Foundation, “System Redundancy with WirelessHART”, *HCF LIT-128*, *Revision 1.0*, June30, 2008.

[5] HART Communication Foundation, “Peer-to-Peer Communication with Wireless HART”, *HCF LIT-129*, *Revision 1.0*, September 5, 2008.

[6] M. Kostadinovic, “Application of Hart protocol for communication needs in process industry”, *Proceedings of INFOFEST Conference*, Budva, Montenegro, 2008. (in Serbian)

[7] M. Kostadinović, M. Stojčev, Z. Bundalo, D. Bundalo: “Control of WirelessHART network”, *Proceedings of INFOTEH Conference*, Jahorina, Bosnia and Herzegovina, 2009. (in Serbian)

[8] M. Kostadinović, M. Stojčev, Z. Bundalo, D. Bundalo: “Design, Implementation and Simulation of WirelessHart Network”, *Proceedings of TELSIKS Conference*, Niš, Serbia, 2009.

[9] M. Kostadinović, M. Stojčev, Z. Bundalo, D. Bundalo: “Application of modified TrueTime simulator in control of WirelessHart networks”, *Proceedings of Intrenational Symposium Ee 2009*., Novi Sad, Serbia, October, 2009.