OPC COMMUNICATION IN REAL TIME

M. Mrosko, L. Mrafko

Slovak University of Technology, Faculty of Electrical Engineering and Information Technology

Ilkovičova 3, 812 19 Bratislava, Slovak Republic

Abstract

Control of real systems in practice brings many complications. The control algorithm is usually implemented directly in the control system connected to controlled system by using of conventional controllers. In practice at the lower level are most often used control systems based on PLCs (Programmable Logic Controllers). However implementation of modern control methods (adaptive control, predictive control, etc.) includes increased computation demands, therefore the realization of the algorithms in PLCs is difficult or almost impossible. One of the possible solutions is to use other control system, e.g. computer with running control algorithm. In this case we need to provide feasible communication between both control systems. In our case we used the well-known communication standard called OPC (OLE for Process Control) for communication between PC and PLC. In this paper we provide testing of this communication standard with reference to quality of control of real system.

1 PLC and PC in control process

Control of industrial processes is often realized using small digital computers called the programmable logic controllers (PLCs), where the hardware and software are specifically adapted to industrial environment. PLCs represent a cost effective solution for control of complex systems. They offer many advantages, such as flexibility (they can be reapplied to control other systems quickly and easily) and reliability (immunity to electrical noise, resistance to vibration and impact). However, this type of controllers usually offers only simple control structures, such as on-off control or PID control loops. [2] However implementation of modern control methods (adaptive control, predictive control, etc.) includes increased computation demands, therefore the realization of the algorithms in PLCs is difficult or almost impossible. One of the possible solutions is to use other control system, e.g. PC with running control algorithm. In this case we need to provide feasible communication between both control systems. In our case we used the well-known communication standard called OPC (OLE for Process Control) for communication between PC and PLC. In this paper we provide testing of this communication standard with reference to quality of control of real system.

2 OPC

OPC (originally OLE for Process Control) which stands for Object Linking and Embedding (OLE) for Process Control, is an industrial standard created with the collaboration of a number of leading worldwide automation hardware and software suppliers, working in cooperation with Microsoft. The standard is maintained by OPC Foundation [4] and widely used within the industrial automation to facilitate the interoperability of control devices from different manufacturers. It specifies the mechanism for communication of different data sources and client applications within the process control. The data source can be a process control system, a database or a supervisory control application. Reference [3] gives detailed information about OPC, and how OPC can be beneficial for research and development and presents an overview of the latest developments and standards.

The OPC Specification is a non-proprietary technical specification that defines a set of standard interfaces based upon Microsoft's OLE/COM/DCOM platform and .NET technology. The application of the OPC standard interface enables the interoperability between automation / control applications, field systems / devices and business / office applications.

Traditionally, each software or application developer was required to write a custom interface or server/driver, to exchange data with hardware field devices. OPC eliminates this requirement by defining a common, high performance interface that permits this work to be done once, and then easily reused by HMI, SCADA, Control and custom applications.

OPC server is the software application which operates as the application programming interface (API) or as the protocol converter. OPC Server is connected to a device such as PLC, distributed control system (DCS), remote terminal unit (RTU) or the data source (database or user interface) and translates the data into a standard-based OPC format.

OPC compliant applications such as a human machine interface (HMI), historian, spreadsheet, trending application, etc. can be connected to the OPC Server and then they can use it to read and write the device data. The OPC Server is based on a Server/Client architecture.

MATLAB® (MATLAB Help) is a high-level language and interactive environment that enables to perform computationally intensive tasks. It is often used in academic community for design, analysis and simulation of advanced control techniques.

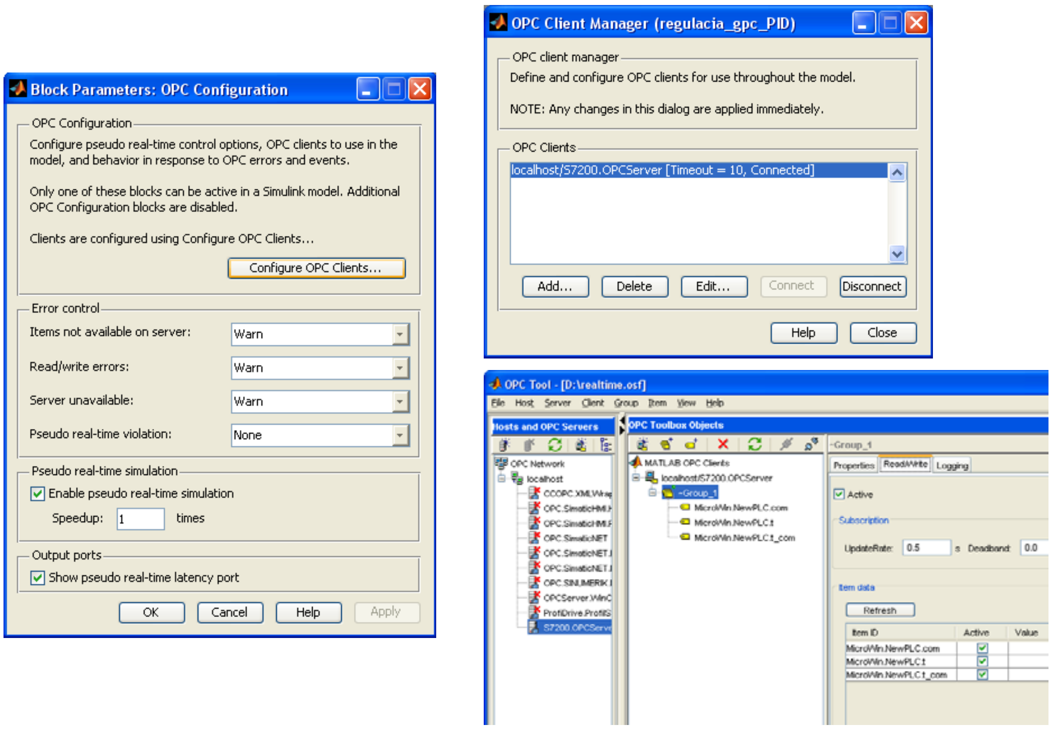
The OPC Toolbox™ (MATLAB Help) extends MATLAB® and Simulink® with tools for interacting with OPC servers. It enables to read, write, and log OPC data from devices that conform to the OPC Foundation Data Access standard, such as distributed control systems, supervisory control and data acquisition systems, and programmable logic controllers. The toolbox enables MATLAB and Simulink products to respond to an OPC server- or OPC Toolbox software-initiated event, such as a shutdown, server error, or item value change. MATLAB with OPC Toolbox can be used in process industries for data analysis, visualization, simulation, and rapid prototyping of algorithms on real processes.

The OPC Toolbox provides three ways to implement an OPC Data Access Client:

1. Execute all OPC Toolbox functions directly from the MATLAB command line or incorporate them into the MATLAB applications.
2. Use the graphical user interface (GUI) to rapidly connect to OPC servers, create and configure OPC Toolbox objects, and read, write, and log data.
3. Use the Simulink Blockset library to read and write data to and from the OPC server while simulating a system.

When used in MATLAB [1], the toolbox employs an intuitive, hierarchical object structure to help you manage connections to OPC servers and collections of server items or tags. You create an OPC Data Access Client object to connect to an OPC server. This connection lets you browse the server name space and retrieve properties of each item stored on the server. You create Data Access Group objects to control sets of Data Access Item objects, which represent server items. The toolbox allows configuring and controlling all client, group, and item objects by modifying their properties. OPC Tool shown in Figure 1 enables to browse the server's name space, configure the objects, and read and write the OPC data. It also enables to log OPC data into MATLAB for analysis and plotting.

Simulink's OPC toolbox offers a configuration block to specify the OPC clients used in the model, to define the behavior for OPC errors and events and to set the real-time behavior. During the simulation, the model executes in pseudo real-time, matching the system clock as closely as possible by automatically slowing the simulation. The block parameters can also be configured so that the simulation runs more slowly than the system clock. OPC Configuration window and the window for OPC Client management can be seen in Figure 1.


Block Parameters: OPC Configuration

OPC Configuration

Configure pseudo real-time control options, OPC clients to use in the model, and behavior in response to OPC errors and events.

Only one of these blocks can be active in a Simulink model. Additional OPC Configuration blocks are disabled.

Clients are configured using Configure OPC Clients...

Configure OPC Clients...

Error control

Items not available on server:	Warn
Read/write errors:	Warn
Server unavailable:	Warn
Pseudo real-time violation:	None

Pseudo real-time simulation

☑ Enable pseudo real-time simulation

Speedup: 1 times

Output ports

☑ Show pseudo real-time latency port

OK Cancel Help Apply

OPC Client Manager (regulacia_gpc_PID)

OPC client manager

Define and configure OPC clients for use throughout the model.

NOTE: Any changes in this dialog are applied immediately.

OPC Clients

localhost/S7200.OPCServer [Timeout = 10, Connected]

Add... Delete Edit... Connect Disconnect

Help Close

OPC Tool - [D:\realtime.osf]

File Host Server Client Group Item View Help

Hosts and OPC Servers

OPC Network
localhost
CCOPC.XMLWrap
OPC.SimaticHMI.H
OPC.SimaticHMI.F
OPC.SimaticNET
OPC.SimaticNET.I
OPC.SimaticNET.I
OPC.SINUMERIK.I
OPCServer.WinC
ProfiDrive.ProfiS
S7200.OPCServ

OPC Toolbox Objects

MATLAB OPC Clients
localhost/S7200.OPCServer
~Group_1
MicroWin.NewPLC.com
MicroWin.NewPLC.t
MicroWin.NewPLC.t_com

~Group_1

Properties Read/Write Logging

☑ Active

Subscription

UpdateRate: 0.5 s Deadband: 0.0

Item data

Refresh

Item ID	Active	Value
MicroWin.NewPLC.com	✓	
MicroWin.NewPLC.t	✓	
MicroWin.NewPLC.t_com	✓	

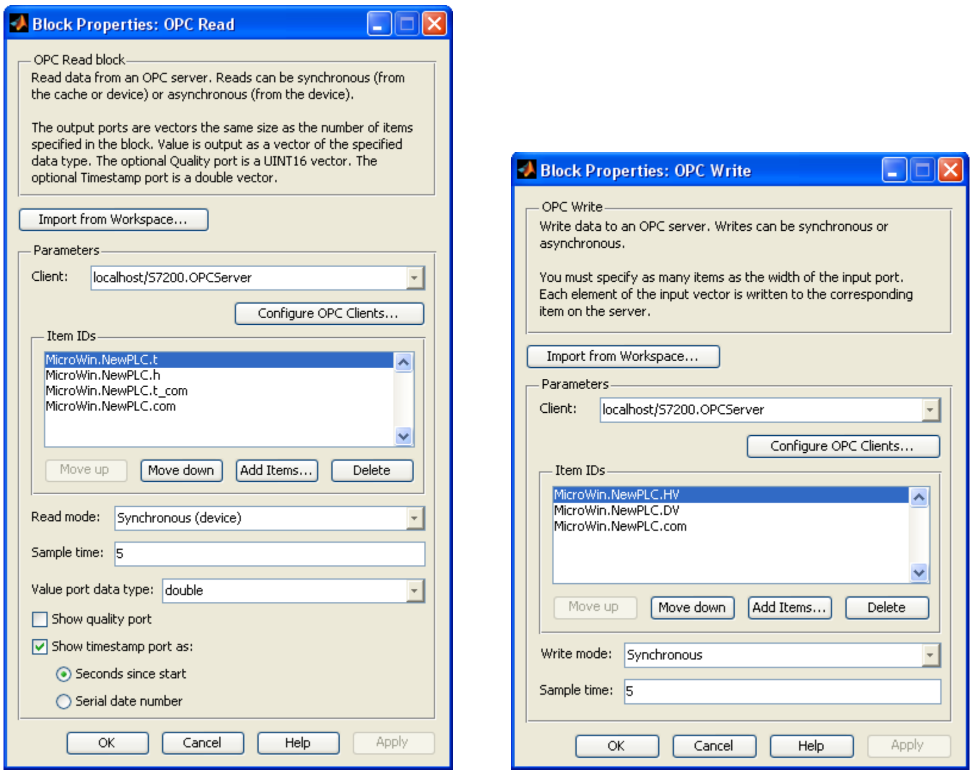
Figure 1: OPC Configuration, OPC Client Manager, OPC Tool

In the OPC Configuration window, *Pseudo real-time simulation* panel allows configuring options for running the simulation in pseudo real-time. When checkbox "Enable pseudo real-time simulation" is checked, the model execution time matches the system clock as closely as possible by slowing down the simulation appropriately. The "Speedup" setting determines how many times faster the simulation runs comparing to the system clock. For example, when "Speedup" is set to 2, it means that a 10-second simulation will take 5 seconds to complete.

Note that the real-time control settings do not guarantee real-time behavior. If the model runs slower than real time, a pseudo real-time latency violation error occurs. You can control how Simulink responds to a pseudo real-time latency violation using the settings in the *Error control* panel. The "Show pseudo real-time latency port" check box enables to output the model latency. When it is checked, the pseudo real-time latency (in seconds) appears as an output port of the "OPC Configuration" block. Pseudo real-time latency is the time spent by waiting for the system clock during each step. If this value is negative, the simulation runs slower than real time and the Simulink action is determined in the "Pseudo real-time violation" setting.

Once a group object containing item objects is created, you can read from or write to individual items or all the items in the group simultaneously. In MATLAB, read and write operations can occur either synchronously (MATLAB execution is blocked until the operation is complete) or asynchronously (MATLAB can continue processing while the operation is in progress).

In Simulink, the read and write blocks retrieve and transmit data synchronously or asynchronously to and from the OPC server. The blocks contain a client manager that makes possible to specify and manage the OPC server, select items, and define block sample times. The OPC Read Block shown in Figure 2 enables to choose items from the OPC server and to read online plant data directly to the Simulink model. The OPC Write Block (Figure 2) enables to choose items from the Simulink model and to write online plant data directly to the OPC server.


Block Properties: OPC Read

OPC Read block

Read data from an OPC server. Reads can be synchronous (from the cache or device) or asynchronous (from the device).

The output ports are vectors the same size as the number of items specified in the block. Value is output as a vector of the specified data type. The optional Quality port is a UINT16 vector. The optional Timestamp port is a double vector.

Import from Workspace...

Parameters

Client: localhost/S7200.OPCServer

Configure OPC Clients...

Item IDs

MicroWin.NewPLC.t
MicroWin.NewPLC.h
MicroWin.NewPLC.t_com
MicroWin.NewPLC.com

Move up | Move down | Add Items... | Delete

Read mode: Synchronous (device)

Sample time: 5

Value port data type: double

☐ Show quality port

☑ Show timestamp port as:

● Seconds since start

○ Serial date number

OK | Cancel | Help | Apply

Block Properties: OPC Write

OPC Write

Write data to an OPC server. Writes can be synchronous or asynchronous.

You must specify as many items as the width of the input port. Each element of the input vector is written to the corresponding item on the server.

Import from Workspace...

Parameters

Client: localhost/S7200.OPCServer

Configure OPC Clients...

Item IDs

MicroWin.NewPLC.HV
MicroWin.NewPLC.DV
MicroWin.NewPLC.com

Move up | Move down | Add Items... | Delete

Write mode: Synchronous

Sample time: 5

OK | Cancel | Help | Apply

Figure 2: OPC Read and OPC Write properties

The toolbox enables to log data as it changes over time. Data can be logged to a memory or to a disc. The MATLAB and add-on toolboxes can then be used to analyze and visualize the data, to design the control systems or to optimize the plant performances.

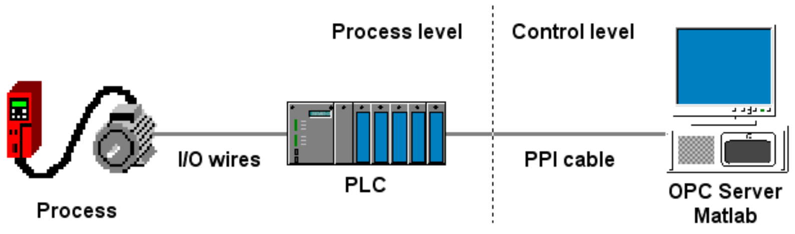


Figure 3: Control system with OPC communication

During the simulation of real-time (or pseudo real-time) control strategies in Simulink, an issue of block execution order can occur. Without explicitly defining block priorities in Simulink, the blocks are executed in order given by so called “sorted order”. There are two basic rules, which affect the block order:

1. each block, which drives another block's “direct-feedthrough” ports must precede this block,
2. blocks without direct-feedthrough ports must give precedence to any blocks that drive the direct-feedthrough ports.

Direct-feedthrough port is an input port, whose current value determines current value of one of the block's outputs. Examples of blocks that have direct-feedthrough ports include the Gain, Product, and Sum blocks. Examples of blocks that have non-direct-feedthrough inputs include the Integrator block (its output is a function purely of its state), the Constant block (it does not have an input) and the Memory block (its output is dependent on its input in the previous time step).

In Simulink there is also a possibility to assign block's priority, by which it is possible to influence the blocks sorted order. Block's priority can be assigned interactively using the “Priority” field of the block's “Block Properties” dialog box or programmatically using “set_param” command. The lower the number, the higher the priority; that is, 2 means the higher priority than 3. Higher priority blocks appear before lower priority blocks in the sorted order, though not necessarily before blocks that have no assigned priority.

3 Case study

During the implementation of real-time experiments in Simulink through the OPC communication it can happen that the real execution time is longer than the simulation time set in the Simulink configuration parameters. Figure 4 shows the results of the real-time simulation where the real execution time of 30 s simulation was 36.5 s. The latency at each step is negative, which means that each sample period is longer than 1 s.

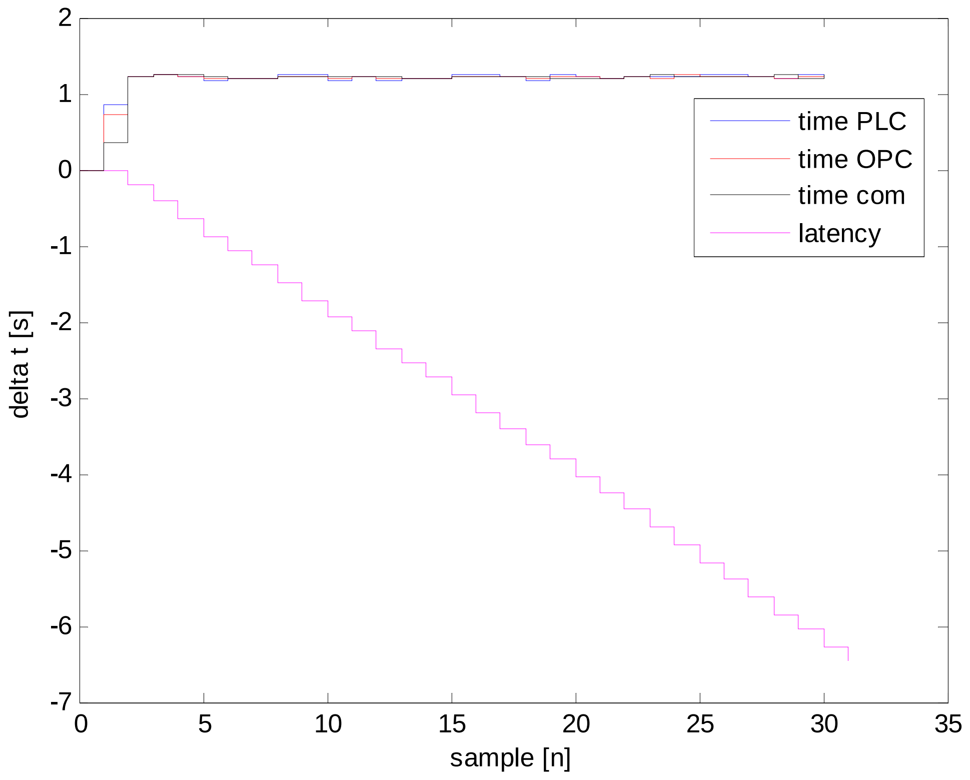


Figure 4: Sample time and latency

Latency represents the sample time margin, the larger latency means larger margin, i.e. the sample time can be more decreased. There are several factors that allow getting the execution time near the value set in the Simulink scheme.

The speed of data transfer across the network is given by the baudrate, which is typically measured in units of kilobaud (kbaud) or megabaud (Mbaud). The baudrate measures how much data can be transmitted within a given amount of time. Latency of samples with different baud rate is shown in Figure 5.

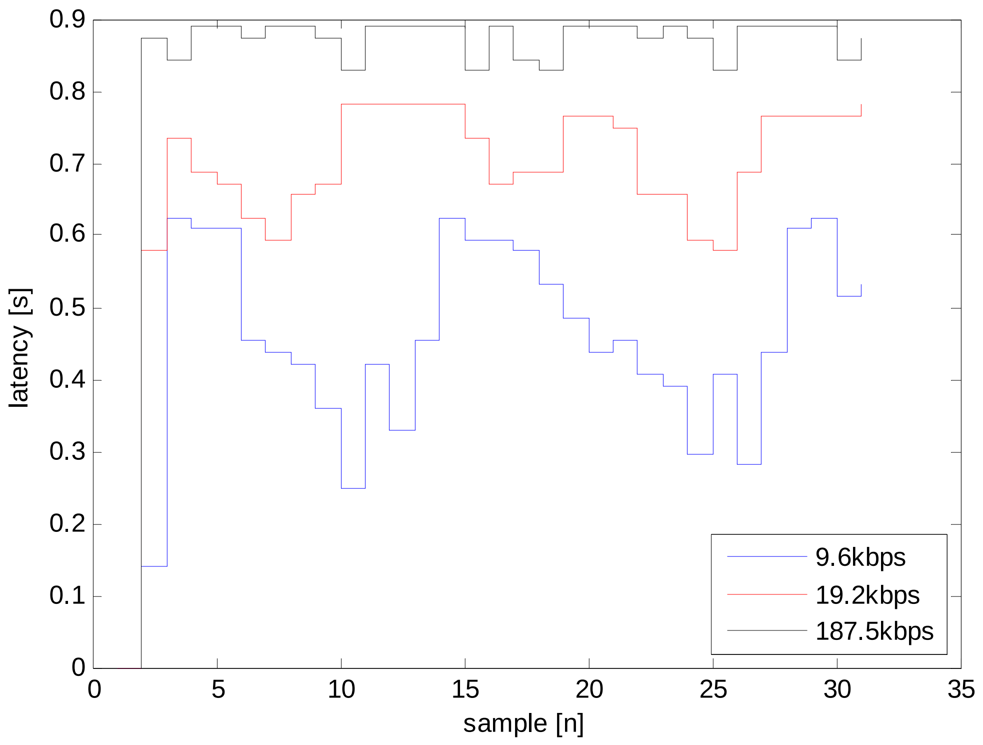


Figure 5: Latency dependence on baudrate

Latency also depends on the number of OPC Read and OPC Write blocks as shown in Figure 6. It can be seen that the number of OPC Write blocks influences the latency more than the number of OPC Read blocks.

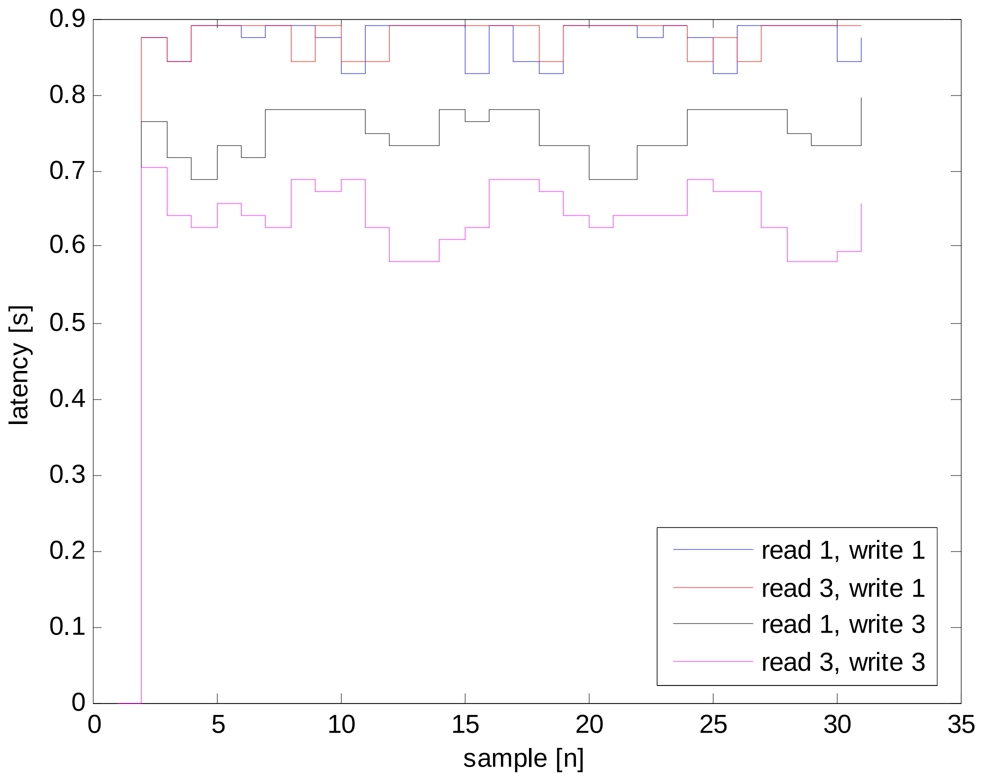


Figure 6: Latency dependence on the number of OPC Read and Write blocks

In Figure 7 dependence of latency on the number of OPC Read and OPC Write items is investigated. It can be concluded, that the number of OPC Read and Write items does not significantly influence the latency.

When implementing the experiments with real processes it is important to properly set the communication parameters, more specifically, the data transfer rate, which is often forgotten in practice. Another very important factor is the sample time, which should be chosen taking into account the number of communication items (the number of exchanged process and control variables) and the sufficient computational reserve in case of the incidental processor overload by another process.

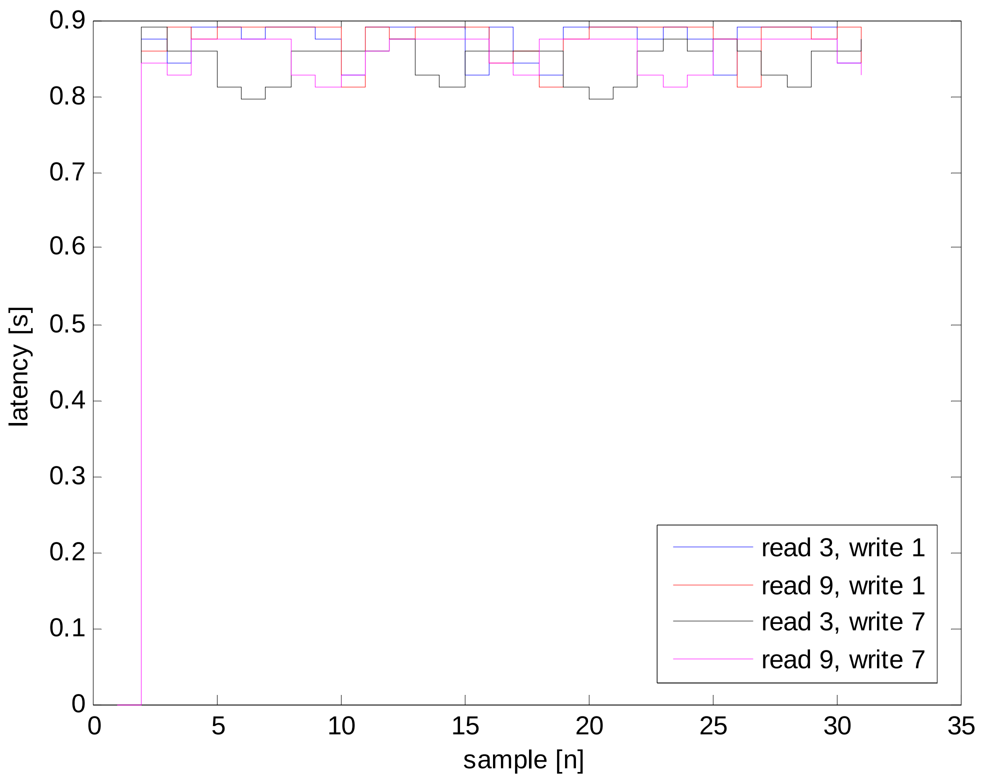


Figure 7: Latency dependence on the number of OPC Read and Write items

Implementation of real-time control strategies in Simulink and OPC communication meets some restrictions that are critical for processes with fast dynamics. The order of blocks execution in Simulink is highly influenced by the time difference between the sample reading and writing. It is essential to analyze the latency and the sample time variance, which is often omitted in setting the pseudo-real time properties in Simulink. This allows avoiding the problems in implementation of real-time control systems.

Conclusion

In this paper communication standard OPC (OLE for Process Control) for communication between PC and PLC with reference to quality of control of real system has been tested. Latency represents “reserve” for sample time. Higher actual latency means higher reserve – the more can we lower actual sample time. By real-time simulations with connection to real-life systems it is needed to take care of the correct communication settings, especially to use optimal baudrate, which is often in praxis forgotten. As in real-life control also by using Simulink there is an important issue – correct selection of sample time. By properly chosen sample time accordingly to number of needed communication objects (number of transferred process values and manipulated values) and also to computational needs we can keep the latency high enough to have a reserve also for possible occasional higher CPU loads caused by another processes. Testing or implementation of real-time control strategies using Simulink and OPC communication indeed meets some restrictions – it is hardly realizable by fast controlled processes. There is a need to analyze the measured latency to check the sample time dispersion and sample time reserve. These issues are often forgotten by using (pseudo) real-time properties of Matlab/Simulink for real time control.

Acknowledgements

This work has been supported by the Slovak Scientific Grant Agency, Grant No. 1/2256/12 and by the Slovak Research and Development Agency under the contract APVV-0211-10.

References

[1] MATLAB Help (2007), MATLAB OPC Toolbox Help, The Mathworks. Available: http://www.mathworks.com
[2] Miklovičová, E. and Mrosko, M. (2010), Implementation of Predictive Control on Industrial Controllers, AT&P Journal Plus, pp. 39-43.
[3] Schwarz, M. H. and Boercsoek, J. (2007), A Survey on OLE for Process Control (OPC), Proc. of the 7th WSEAS International Conference on Applied Computer Science, Venice, Italy, pp. 186-191.
[4] Specifications of OPC, OPC Foundation 1998-2010. Available: http://www.opcfoundation.org

Marián Mrosko
Slovak University of Technology, Faculty of Electrical Engineering and Information Technology, Institute of Control and Industrial Informatics, marian.mrosko@stuba.sk,

Leo Mrafko
Slovak University of Technology, Faculty of Electrical Engineering and Information Technology, Institute of Control and Industrial Informatics, leo.mrafko@stuba.sk