RESEARCH PAPERS
FACULTY OF MATERIALS SCIENCE AND TECHNOLOGY IN TRNAVA
SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

2021 Volume 29, Number 49

VIRTUAL CONTROL OBJECT FOR SIEMENS LOGIC CONTROLLERS AND ONLINE PLC EDUCATION

Michal BILČÍK[1], Maximilián STRÉMY[2], Dominika JANÍKOVÁ[2], Dušan HORVÁTH[2]

SLOVAK UNIVERSITY OF TECHNOLOGY
FACULTY OF MATERIALS SCIENCE AND TECHNOLOGY
[1]INSTITUTE OF APPLIED INFORMATICS, AUTOMATION AND MECHATRONICS,
[2]ADVANCED TECHNOLOGIES RESEARCH INSTITUTE
ULICA JÁNA BOTTU 2781/25, 917 24 TRNAVA, SLOVAK REPUBLIC
e-mail: xbilcikm@stuba.sk, maximilian.stremy@stuba.sk, dominika.janikova@stuba.sk, dusan_horvath@stuba.sk
Received 7 May 2021, Accepted 13 May 2021, Published 24 November 2021

Abstract

This paper deals with the possibility of creating virtual C # models connectable to a virtual PLC of Siemens. The models will lead to the improvement and simplification of the online PLC education. Hardware solutions cost a lot of money, while using a software version of PLC leads to cheaper simulations. Paper presents the design and implementation of the virtual models, including the ways of the interconnection with PLCSIM, visualization and finally demonstration for the selected virtual model.

Keywords

Siemens PLC, virtual object, online education, control

INTRODUCTION

In the pandemic times, the distance education becomes more and more frequent [1]. In addition to number of advantages, online education also has certain disadvantages. One of the disadvantages is that demonstrations of practical things are not always presented in online education. The benefit of virtual objects is the improvement and simplification of online education for teachers and, at the same time, for students. The aim was to create the virtual control objects that would support and facilitate the online education of Programmable logic controllers (PLC), using theoretical and practical knowledge. PLCs (also known as programmable controllers) are computational devices constructed for an industrial use. They are designed to monitor process (reading inputs) and to control outputs to automate the process, by making decision based on stored program [2]. Some available software solutions give us possibility to simulate a process and create a control program for PLC without spending money for purchasing hardware components. In the case of Siemens products, we can use PLCSIM v5.4 [3, 4]. We are able to “construct” a virtual process, which can be easily modified and does not cost money. The virtual process includes virtual objects. In our case, we used C# programming language to create the virtual objects, since it is simple, modern, object-oriented, based on the concepts from several languages, such as Java programming language [5].

In the research described in this article, we analysed the possibilities of creating virtual objects, including system requirements, subsequent design, implementation and testing possible solution for Siemens PLC.

FEASIBILITY STUDY OF CREATING VIRTUAL MODELS CONNECTABLE TO PLCSIM

Virtual models connectable to PLCSIM can be implemented in various languages supporting the object-oriented programming (C ++, C #, Java, Python) and in modelling and simulation environments such MATLAB as well (for example in Stateflow). The following subchapters will explain two options for connecting a virtual model with PLCSIM in the C # programming language.

Virtual models in the C # programming language connected via S7.Net

One of the options for connecting the virtual model and the PLCSIM program is to use the **S7.NET** driver. The driver is written in C #, which means the programmer does not have to control any interoperability with native code, but simply uses object-oriented programming and all the features of .NET. [6] However, the controller only works with a Siemens PLC connected via Ethernet. The condition is that the PLC must have a Profinet CPU or Profinet communication module, compatibility with S7-200, S7-300, S7-400, S7-1200 and S7-1500. [7]

Virtual models in the C # programming language connected via S7ProSim V5.4

Another way is to use S7ProSim as an interface component, which will be served for manipulation with PLCSIM data from the user program without the need of an Ethernet connection. [8] It might be used in any application that can accept COM objects to connect with a simulation process in S7-PLCSIM. [9]

Add a declaration class ProSim to project

The class “Form” is extended by adding a new object PLC; it is implemented as follows:

```
public partial class Form : Form
   {
   public S7PROSIMLib.S7ProSimClass PLC = new S7PROSIMLib.S7ProSimClass();
   } [10]
```

For S7ProSim, it is not necessary to specify the type of PLC, IP address, rack and position of CPU. In this way, connection and communication with the PLC is simplified compared to the use of a S7.Net controller.

Read and write bytes from PLC

To read the output variables from the Step7 program, the combination of the timer, constantly checking the specified output address, and the ReadOutputPoint method is utilized. The function for reading the output variable at the address Q1.2 takes the form:

```
private void timer_CitanieVystupov_Tick(object sender, EventArgs e)
{
object Q1_2 = true;
PLC.ReadOutputPoint(1, 2, S7PROSIMLib.PointDataTypeConstants .S7_Bit, ref Q1_2);
} [11]
```

WriteInputPoint method is used for writing the change of input variables state to Step 7 - PLCSIM, by using the checkBox as follows:

```
private void checkBox0_5_Vstup_CheckedChanged(object sender, EventArgs e)
{
object I0_5 = checkBox0_5_Vstup.Checked;
PLC.WriteInputPoint(0, 5, ref I0_5);
} [11]
```

The first number, in parentheses of method ReadOutputPoint, represents the starting byte position in the peripheral image buffer to read. The second number represents the Bit position (in bytes). In the above example, it is the output of Q1.2 and therefore the first number is 1 and the second is 2. WriteInputPoint works similarly, with the difference that, instead of reading, it is written and the input variable I0.5 is used.

Table 1 Functional user requirements catalogue

Functional requirements		**Additional information**
R01	Virtual model shall be able to connect with PLCSIM program.	-
R02	Virtual model shall be able to read CPU status and scan type.	After successful connection, the virtual model will permanently read the current CPU status (RUN_P, RUN, STOP) and scan type (Single Scan or Continuous).
R03	Virtual model shall be able to set up the CPU status using relevant buttons.	While the model is connected to the PLC, the model will change the state of the CPU (RUN_P, RUN, STOP) depending on the pressing of the adapted buttons and thus control the PLC. Therefore, it will not be necessary to work in the PLCSIM for PLC control.
R04	Virtual model shall be able to set up input address using relevant buttons.	The number of buttons and input addresses will vary depending on the needs of the virtual model.
R05	Virtual model shall be able to show its current status.	Status for traffic light depends on the currently active input address - "Day shift", "Night shift". Status for conveyor depends on the running of the motor at address Q4.0 and the sensor at address I1.5 - "Stand-by", "In motion".
R06	Virtual model shall be able to simulate function of object.	According to the program that will be loaded in the PLCSIM, the model will simulate: - traffic light operation and will display the respective colours and their combinations. The model will respond to the states of defined output addresses, then it will simulate any colour combinations and their time intervals, or turn off the traffic light; - operation of the conveyor and display the movement of the package on the conveyor in the forward direction, or the package is located at the end of the belt.

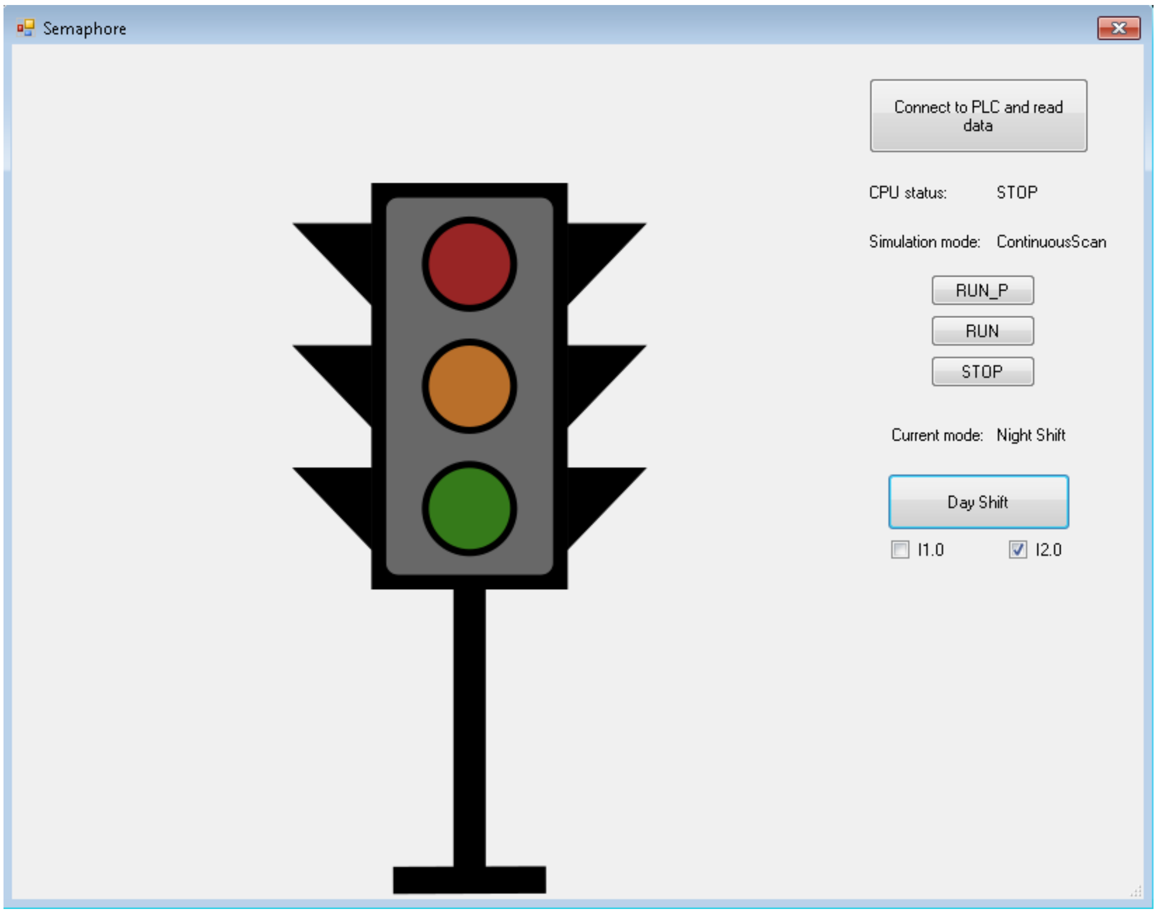
DESIGN AND IMPLEMENTATION

The traffic light and conveyor were designed and implemented as the virtual control models suitable for the beginners in the field of PLC control education.

The virtual traffic light model

Fig. 1 presents an application window called “Semaphore”. On the right side, there are the controls of application. An error message will pop up for the case when the user wants to test the functionality of the remaining buttons, even though user is not connected to PLCSIM using the button – “Connect to PLC and read data”. After successfully logging in to the PLCSIM application, it is no longer necessary to work in the PLCSIM and it is sufficient to minimize it. It is very important to keep PLCSIM running because of data exchange between application “Semaphore” and PLCSIM.

The buttons RUN_P, RUN and STOP represent the buttons in the PLCSIM, and are used to start and stop the PLC module. In the CPU status line, there is information about the current processor status of the used PLC module. Simulation mode line can show SingleScan or ContinuousScan, depending on the selected scan mode in PLCSIM.



CPU status: STOP

Simulation mode: ContinuousScan

Current mode: Night Shift

__Fig. 1__ The application window of virtual traffic light model

The button, located at the bottom of the virtual model, may be named Night Shift and Day Shift depending on the current traffic light mode. There is no information in the Current mode line before first pressing the button. After pressing the button, the information - Night mode is displayed in the Current mode line, input I2.0 switches to the value of logic 1 and the button takes the name Day shift. After pressing the button again, the current traffic light mode changes to - Day shift, input I2.0 switches to the value of logic 0 and I1.0 to the value of logic 1, the button takes the name Night shift again. An example of such use can be seen in Fig. 2.

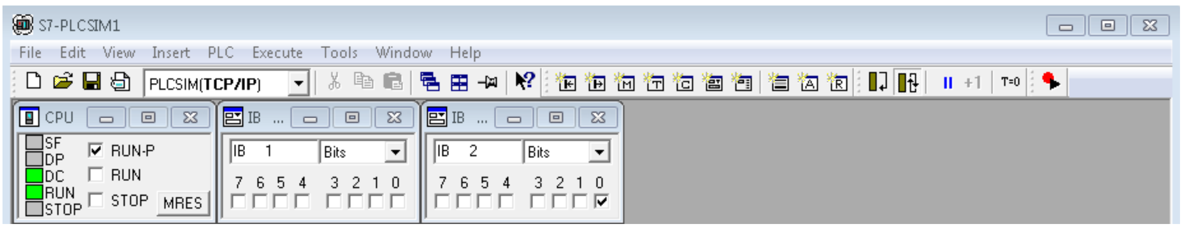


***Fig. 2** Change of input address values performed using a virtual model*

Input variables:

- Day shift – I1.0
- Night shift – I2.0

Output variables:

- Red colour – Q1.0
- Orange colour – Q2.0
- Green colour – Q3.0

The virtual conveyor model

The part of the conveyor user program together with the current status of the inputs/outputs is shown in Fig. 3.

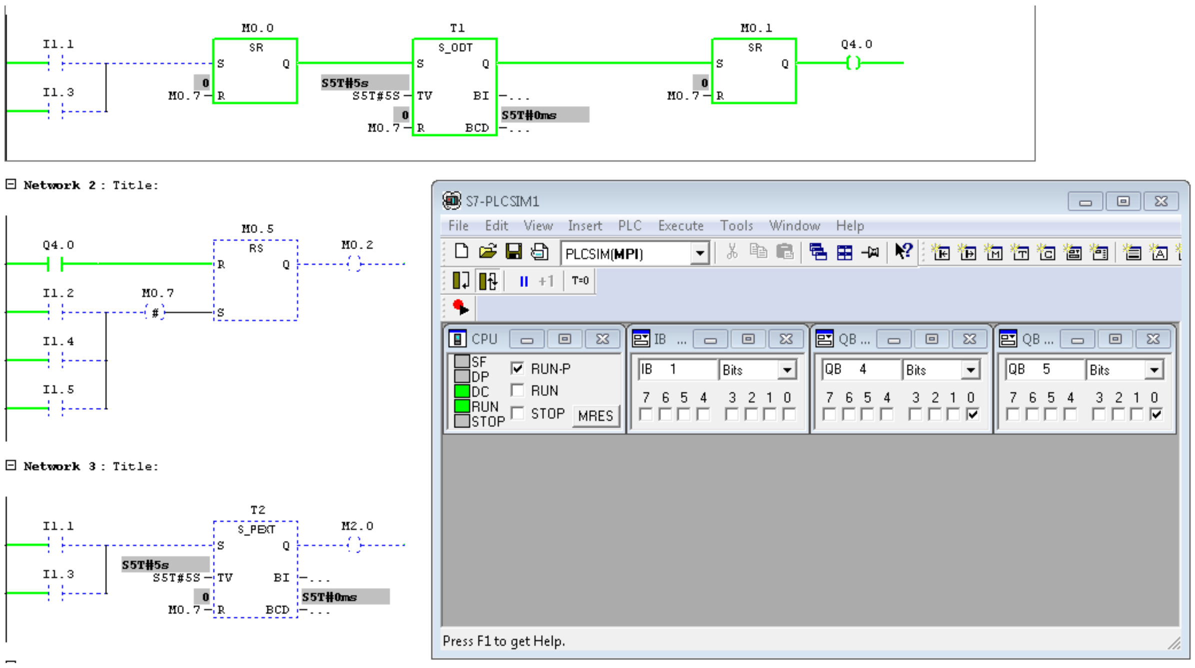


***Fig. 3** Part of the conveyor user program with PLCSIM*

The application window, called “Conveyor 2”, demonstrates the correct functionality of conveyor in case of pressing and releasing the Start1 button (Fig. 4). On the right side, there are application controls, buttons RUN_P, RUN, STOP, Connect to PLC, Read data and lines, CPU status and Simulation mode. The buttons work on the same principle as in the case of the virtual traffic light model. The Start 1, Start 2, Stop 1, Stop 2 and Sensor buttons are used to start or stop the conveyor by changing the values of input addresses I1.1 and I1.3 (for Start 1 and Start 2), I1.2 and I1.4 (for Stop 1 and Stop 2). In the case of the I1.5 sensor, there is situated only a checkbox, which is used to check whether the sensor is compressed by the package or not.

The conveyor responds to the start of motor. The variable of the motor is located at the address Q4.0. After starting the motor, the green arrow under the conveyor belt lights up. The traffic light on the left also responds to the user program, regardless of the correct working. The next step is the animation of package crossing from the beginning of the strip through its centre to the subsequent end of the belt. The length of the animation is generated automatically using a Gaussian random number generator in the range of 1-4 seconds. When the package arrives to the end of the belt and presses the sensor, the package disappears after 2 seconds. The time intervals and the disappearance of the bale are used to simulate the operation of the conveyor. In the line of the conveyor status, different information depending on the current status, similar to conveyor 1 can be displayed. The information "*In motion*" is displayed - in case the Q4.0 motor is running. The information *"Stop"* is displayed - in case Q4.0 motor is inactive. When the sensor I1.5 is pressed, the package is displayed on the right side of the conveyor belt, and the information is displayed in the line Conveyor status as follows:

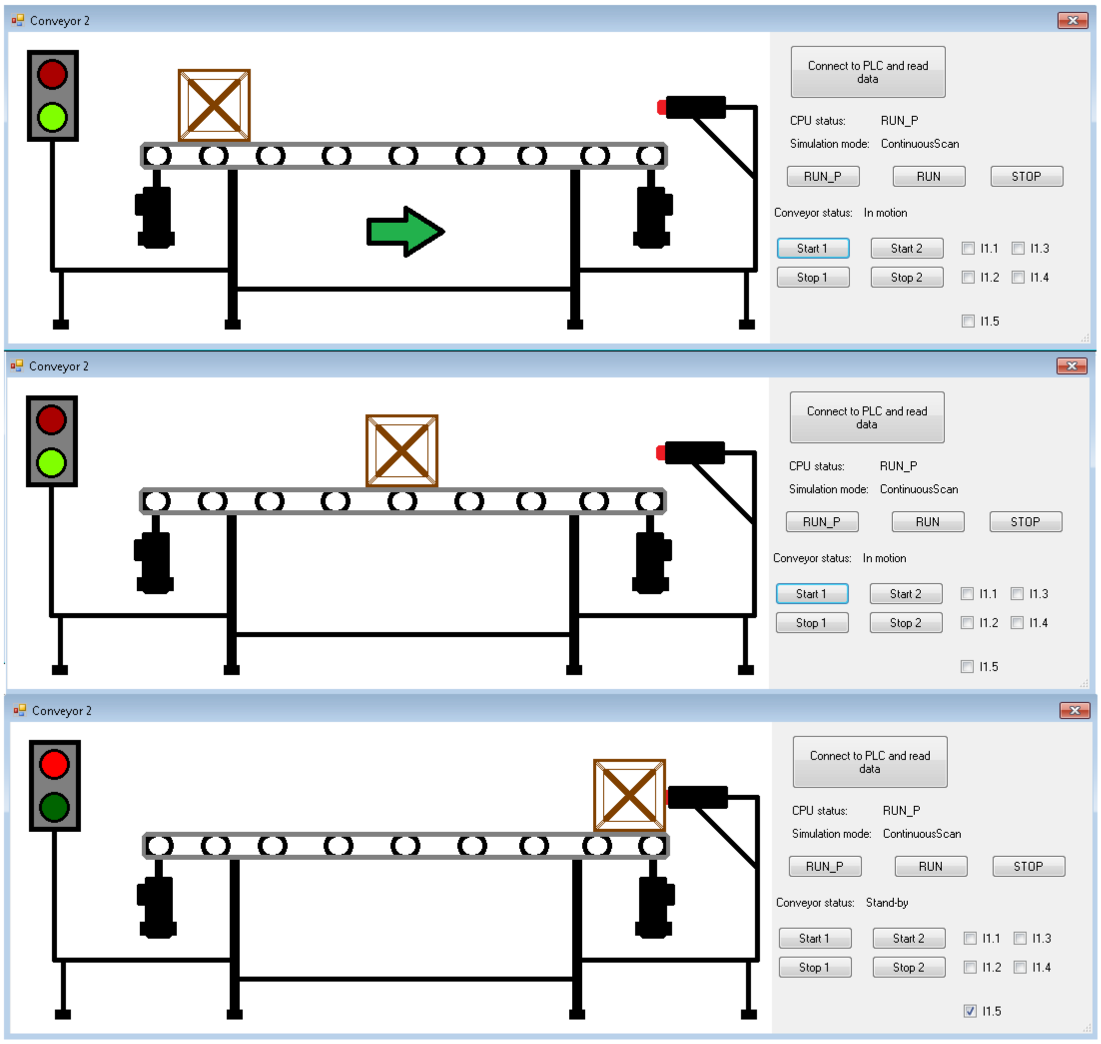
- *"In motion"*: the part is at the end of the belt and there is a risk of collision – in case Q4.0 is still logical 1.
- "S*tand-by* ": the part is at the end of the belt – in case Q4.0 is logical 0.

Setup of input variables:

- Start 1 – I1.1
- Start 2 – I1.3
- Stop 1 – I1.2
- Stop 2 – I1.4
- Sensor – I1.5

Setup of output variables:

- Green colour – Q5.0
- Red colour– Q5.1
- Motor of conveyor – Q4.0



Conveyor 2

CPU status: RUN_P
Simulation mode: ContinuousScan
Conveyor status: In motion

Conveyor 2

CPU status: RUN_P
Simulation mode: ContinuousScan
Conveyor status: In motion

Conveyor 2

CPU status: RUN_P
Simulation mode: ContinuousScan
Conveyor status: Stand-by
[✓ I1.5]

Fig. 4 *Application window of "Conveyor 2"*

CONCLUSION

The virtual models shown in this paper will facilitate and increase effectivity of the PLC online education. The interconnection between the virtual models and the virtual Siemens PLC is implemented by the S7ProSim V5.4 interface component. The virtual model sets the PLC inputs and reads the outputs according to the specified requirements. The state of the outputs is visualized in the form of a simulation - by lighting the desired colour in the case of a virtual model of traffic light, or by simulating the movement of the bale in the case of a virtual conveyor model. Both types of virtual model can be improved by modifying the source code. In the case of a virtual conveyor model, there is opportunity to improve the simulation motion, which may not be able to respond properly to all variations of user code.

Acknowledgement

This publication was supported by the Operational Programme Research and Innovation for the project "Scientific and Research Centre of Excellence SlovakION for Material and Interdisciplinary Research", code of the project ITMS2014+ : 313011W085 co-financed by the European Regional Development Fund.

References

[1] JACOME, E. M., TOAQUIZA, J. F., MULLO, G. M., ANDALUZ, V. H. and VARELA-ALDÁS, J. 2021. “Virtual System for Industrial Processes: Distillation Towers,” in Augmented Reality, Virtual Reality, and Computer Graphics, Cham, 2021, pp. 670–679. doi: 10.1007/978-3-030-87595-4_48.

[2] Siemens Industry, Inc., “Basics of PLCs,” 2016. Accessed: Oct. 29, 2021. [Online]. Available: https://sitrain.us/step/pdfs/version2/Basics_of_PLCs.pdf

[3] Siemens, “Engineering Tools S7-PLCSIM V5.4 incl. SP3,” p. 80, Mar. 2009.

[4] Siemens, “S7-PLCSIM v5.4.8 online Readme,” p. 18, 2017.

[5] M. McGrath, C# Programming in easy steps, 5th ed. 2016. [Online]. Available: https://www.programmer-books.com/wp-content/uploads/2018/08/In.Easy.Steps.Cplusplus.Programming.in.easy.steps.5th.Edition.pdf, ISBN: 978-1-84078-757-3

[6] Communication with Siemens S7 Plc with C# and S7.Net plc driver [Online]. [Accessed: 01-2021] Available at https://www.mesta-automation.com/siemens-s7-plc-c-s7-net-plc-driver/

[7] S7.Net documentation [Online]. [Accessed: 01-2021] Available at https://github.com/S7NetPlus /s7netplus/blob/ master/Documentation/Documentation.pdf

[8] Using S7-PROSIM with Siemens S7-PLCSIM [Online]. [Accessed: 01-2021] Available at https://alexsentcha.wordpress.com/using-s7-prosim-with-siemens-s7-plcsim/

[9] SIEMENS, SIMATIC S7ProSim V5.4 COM Object, User Manual – Edition: 01/2007 [Online]. [Accessed: 01-2021] Available at https://cache.industry.siemens.com/dl/files/855/1139855/ att_29424/v1/S7WSPSCB.pdf

[10] Interface between C# and S7-PLCSIM (Part 2 of 4) [Online]. [Accessed: 01-2021] Available at https://www.youtube.com/watch?v=cOMouM1R0AU&ab_channel=AminRamazanifar

[11] Interface between C# and S7-PLCSIM (Part 4 of 4) [Online]. [Accessed: 01-2021] Available at https://www.youtube.com/watch?v=FRVpULQ7aEE

ORCID

Maximilián Strémy	0000-0003-2918-0714
Dušan Horváth	0000-0003-4138-5966