JIEE Časopis priemyselnej elektrotechniky
Journal of Industrial Electrical Engineering
ISSN 2454-0900
www.jiee.eu

Design and implementation of software for smart glasses in the automation process

[1]*Marek MARCIŠ,* [2]*Tibor VINCE,* [3]*Dawid EWALD,* [4]*Wojciech T. DOBROSIELSKI*

[1,2] Department of Theoretical and Industrial Electrical Engineering , FEI TU of Košice, Slovak Republic
[3,4] Institute of Technology, Kazimierz Wielki University, Bydgoszcz, Poland

[1]marek.marcis@student.tuke.sk, [2]tibor.vince@tuke.sk, [3] dawidewald@ukw.edu.pl, [4] wdobrosielski@ukw.edu.pl

***Abstract* — The purpose of this article is to analyze the technology of smart glasses belonging to the group of wearable devices and their possible use in the automated industry. As the name of the article itself says, this is a design of smart glasses software. It talks about software link between Android OS and Siemens PLC. This link ensures visualization of the control process data as well as its control. The software is designed in Android Studio and then tested on the emulator. Since the application was designed for specific smart glasses, an emulator from the eyeglass maker was used. Emulator is an application on a mobile phone serving as a representation of physical glasses.**

***Keywords* — Android, automation, PLC, Siemens, smart glasses, smartphone.**

I. INTRODUCTION

Thanks to technological advances, we are surrounded by different computers, portable devices, and wireless technologies everywhere in today's world. Most people own a mobile phone so that we have at least one phone, also known as a smartphone. The smartphone can be a phone with a touchscreen that includes many features and applications. Technological equipment is a daily component of mobile workers. Wearable technology represents another stage of development, and these devices shift people's abilities again to a higher level. Wearable devices in general include smart watches and smart glasses.

Currently, there are not many topics or areas that deal with linking the Android operating system with the automation main component, which is a programmable logic controller (PLC). However, are available external libraries that make this connection. Each PLC manufacturer provides both software and hardware to visualize the production process. Without this visualization, machine production where the PLC's controlling element would be meaningless. It serves to display data that needs to be recorded and tracked. As technology is constantly evolving nowadays, such smart glasses can already be used for such visualization. Certainly, it would be a great improvement if operators or supervisors could see the data from the production process directly in front of their eyes without having to operate the static device that visualizes them [1][6].

II. USER REQUIREMENTS FOR SOFTWARE DESIGN

User requirement was defined as follow:

- An appropriate operating system for smart glasses,
- Displaying simple data from the production process,
- Data should serve for reading and simple intervention,
- Compatibility.
- Freely available emulator.

The smart glasses operating system - should include an existing standard OS (Android, MS Windows), which is freely available and has a development environment. They should not be based on the own OS principle. They should be compatible with other smart devices like smartphones.

Simple Data Display - these are data that describes the current production process. Such data is in the PLC that controls the process. The proposed software should be able to retrieve and display the data in an appropriate form.

The data should be read and simple interference – the data should have an informative character for the user, so that the value of the production process is available. Display text and numbers, or simple pictures. The user should also be able to easily intervene in the process, such as turning on and off any part of the production process or devices.

Compatibility - correctly link smart glasses and PLCs that control the automated process. Such linking is provided by external libraries designed to communicate between the smart glasses and Siemens PLC. Another smart part is smart glasses and a smartphone, if smart glasses require the presence of a smartphone. This connection depends on the compatibility of the operating systems of both devices.

Freely available emulator - when designing a software, it would be a good idea to test this program first on an emulator to improve and debug errors. Find a manufacturer that also provides this simulation option to specific glasses.

III. DESIGN SOFTWARE FOR SMART GLASSES

From defined user requirements, it was necessary to select the specific type of smart glasses for which the software will be designed. Several types of smart glasses were available, where just one was selected. Sony SmartEyeglass most fit for user requirements. Requirements for successful implementation of practical output:

Hardware Requirements:

- Smart glasses,
- Smart phone,
- Wi-Fi AP.

Software Requirements:

- Development environment (SDK),
- External library for communication with Siemens PLC,
- Software for PLC programming,
- Smart glass emulator

.**Smart glasses** - the very hardware component is of course smart glasses. However, they are not a necessary part, since the emulator is also sufficient to test and preview the functionality of the software.

Smartphone - serves as a host device for smart glasses. The software architecture is based on applications that run on the smartphone. Of course smartphones and glasses are connected together via Bluetooth or Wi-Fi. The manufacturer also offers a list of compatible mobile devices for smart glasses.

Wi-Fi AP- the condition for communication between the smartphone and the Siemens PLC is that both have to be on the same network. It ensures that the PLC is visible to the smartphone. When they see each other, the information transmission could be established. The proposed application connects to the PLC using an IP address.

The development environment - SmartEyeglass includes the Android operating system. Therefore, the design of the software will be implemented in an environment called Android Studio 3.0.1, which is a freely available development environment for all operating systems like Windows, MacOS and Linux. This whole environment is based on the JAVA programming language principle. It serves for design, testing, and implementation of Android apps.

External library - the Android studio itself does not have the ability to link an application to a PLC. However, there are external libraries that are available and can be secured by this communication. For an Java-based development environment is used an external library called Moka7. Provides communication between Siemens PLC and OS Android [4].

IV. REALIZATION OF SOFTWARE FOR SMART GLASSES

The software should originally have been designed for physical smart glasses that would be used in the production process in automated manufacturing. There are a number of smart glasses on the market, of which only a few are suitable for industrial use, as this technology is in constant development and there is no general standard. The software was designed for specific system.

Description of the observed system.

The software has been designed for a particular system that consists of several parts:
- Siemens PLC S7-300,
- frequency inverters,
- 2 asynchronous and 2 synchronous motors.

Siemens PLC S7-300 - this type of PLC belongs to the group of the most used and most versatile PLCs used in automated production in Europe. In this particular system, it fulfills the management function of the whole system. The control program evaluates signals from the inputs and executes the computation process and then sends the instruction to the outputs to which the actuators are connected. The PLC is connected to the frequency inverters via the PROFIBUS bus. This bus serves to effectively exchange information between the inverter and the PLC. The processor CPU 313C is used to control the entire system. The communication for the PROFIBUS bus is provided by the CP 324-5 communication processor and for the Ethernet it is the CP 343-1 Lean [2].

Frequency inverters - the system contains Siemens drives, namely the SINAMIC S110. The inverters are used to regulation and control the motors. The inverter receives direct commands from the PLC and then sends them the motor via the AS interface used by the actuators. They provide effective engine speed control. Parameters are also monitored as absolute current or motor temperature. The inverters also fulfill a safety function, e.g. protect the motor against the tip of the current. They are powered by a single phase AC voltage of 200-240V. Their power ranges range from 0.12 - 0.75 kW [3].

Motors - motors are generally used to convert electricity to mechanical work. Thus, in the whole system, actuators are formed where they are able to control the mechanisms attached to them. The observed system includes 2 asynchronous and 2 synchronous motors. Asynchronous motors are often used as drives in industrial applications. Synchronously in applications that require exact and constant rotation. In Fig. 1 shows the whole principle of interconnecting the observed system [5].

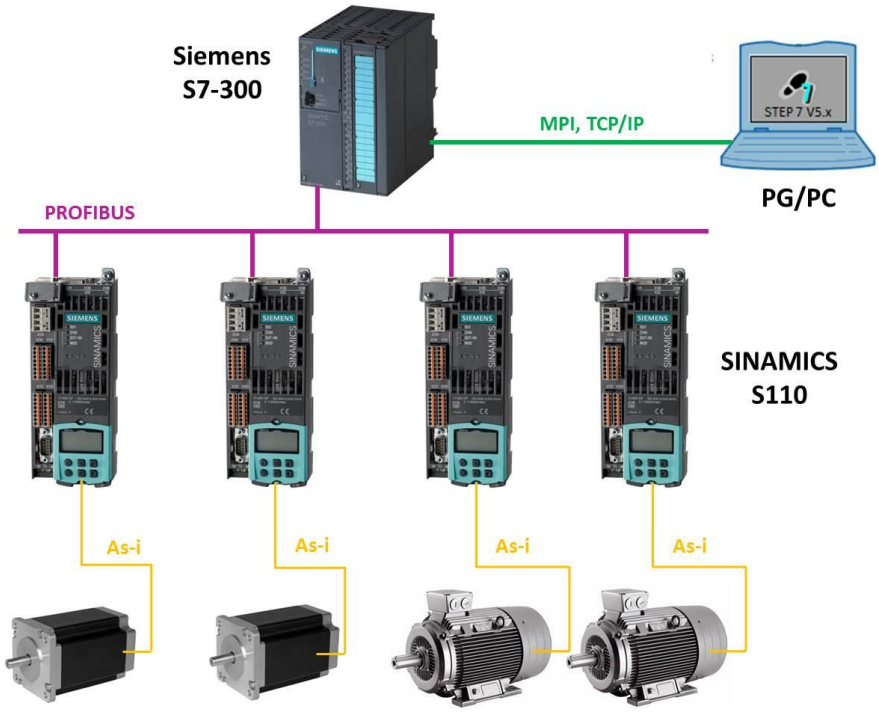

Fig. 1 The principle of linking the observed system

The program is loaded into the PLC by the programing device. This term is a computer that contains the Simatic Manager, software used to program the Siemens PLC.

The control program located in the PLC contains various organizational blocks (OB), data blocks (DB), functions (FC), and function blocks (FBs) to control the whole system. The entire control provides the base block OB1, from which the functions are called and thus the PLC allows control and control of the entire system. However, all important engine information is stored in the DB. Each engine must have its own addresses in memory indicating which DB and its part are in order to avoid conflicts. By analyzing the already created control program, data block addresses have been identified that contain the necessary data such as:

- actual engine speed,
- set engine speed,
- engine temperature,
- state in which the motor is located - off, on,
- absolute current,
- actual torque.

By analyzing the visualization proposed in WinCC Explorer for the observed system, the actual values of the actuators (motors) that are currently defined in this visualization were found. Therefore, in the design of the software it was necessary to choose a suitable conversion factor in order for the application reading values from DB to write the real value of the monitored quantities. The same procedure is maintained for the other measured variables. Exception is to monitor engine status. The "Status" parameter is defined as the binary value:

- Status = 1 - Engine on,
- Status = 0 - Engine off.

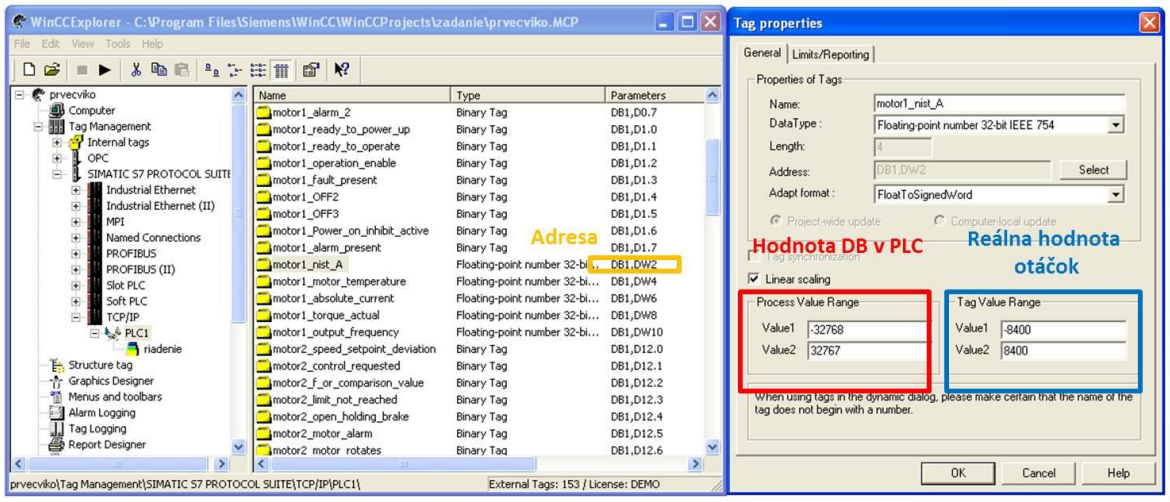

Fig. 2 Comparison of PLC data and real data in WinCC Explorer

All division ratios are shown in **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

Table 1
Numerical representation of monitored data

Monitored variable	Real value		Value in PLC		Division ratio
	min.	max.	min.	max.	
Real speed [rpm]	-8400	8400	-32768	32767	1/3.9
Temperature of engine [°C]	0	200	0	32767	1/163.8
Absolute current [A]	-5.32	5.32	-32768	32767	1/6142.4
Actual torque [N.m]	-7.08	7.8	-32768	32767	1/4628.2
Status [1]	0	1	0	1	0
Requested speed [rpm]	-8400	8400	-32768	32767	1/3.9

SmartEyeglass application mediates communication between the smartphone and the Siemens PLC S7-300, which serves as a controlling member of the entire system. It contains important data about the engines that the app captures and then displays on the mobile phone in the SmartEyeglass emulator. This subchapter describes the program in Android OS and why it is based on individual features.

A. Basic software features

processPicture (CameraEvent event) - this function is used to decode the QR code where the PLC IP address is encoded and the name of the engine we want to track. The entire chain is then subdivided into two sub-chains. One contains an IP address and a second engine name.

connectToPLC () - the function tries to connect to the PLC using the IP address stored in the subprocess from processPicture. If you can connect with the output, it is reported that the PLC is available. If a connection fails, you will not be able to see a specific error message.

- `res = client.ConnectTo(ipAdrress, 0, 2);`

The res variable contains an integer value if it is equal to 0, meaning that the PLC is available and the applications have managed to join it. If it contains a different value, it means that there is an error that appears on the display.

disconnectPLC () - when the function is called, the application disconnects the PLC. It is necessary to call at the end of the program or when changing the monitored motor to avoid conflict.

readVal () - after successful connection to the PLC, the function for reading and displaying data in the emulator is automatically called. It contains predefined addresses that are tracked according to which engine it is. The function that defines a specific address in the PLC memory demonstrates the following example:

- `client.ReadArea(S7.S7AreaDB, 1, 2, 2, data1);`

This part of the source code serves to define the address of 2 bytes (word). Specifically, it is a part of the DB1.DBW2 data block that contains the actual engine speed1.

writeVal () - as the abbreviation of this function itself says, its call is used to write values to the PLC. Turn on and off a particular engine that is currently being watched. It also allows to increase and decrease the engine speed.

- `client.WriteArea(S7.S7AreaDB, 2, 2, 2, str6);`

The WriteArea function ensures that the address to which the desired value is written is correctly defined. This example serves to define the address for the required DB2.DBW2 speed.

B. Connect to PLC

Because the entire software needs to know which PLC to connect to and which engine data to track. This information has to be encoded appropriately. The most convenient way was using QR code.. Currently, there are a variety of online tools on the Internet where you can create such a QR code (eg http://goqr.me/). The encoded information contains two strings separated by a dot***: ip_address_PLC; name_engine***. Engine name: Motor1, Motor2, Motor3 or Motor4. For each engine is created, a specific QR code.

After running the emulator by moving the Touch Sensor step by step, you need to find the application " S7_SSG " in the main menu.

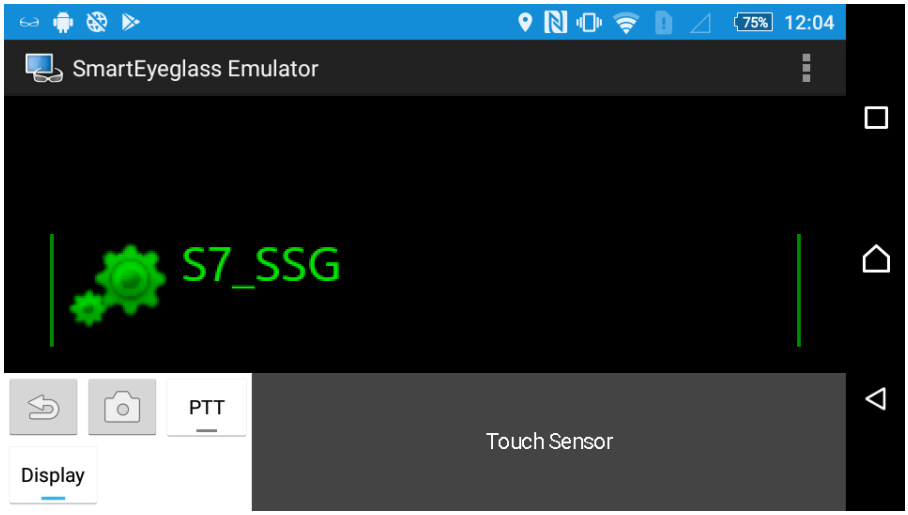

Fig. 3 Application in the emulator main menu

When the application is launched, a message appears prompting the user to press the contact surface of the sensor to capture the QR code.

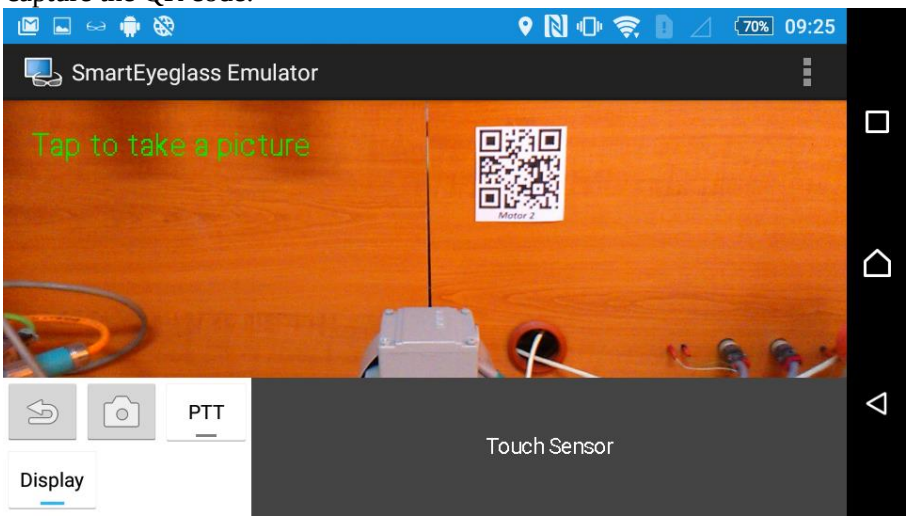

Fig. 4 Startup screen after running the app

It can take a few seconds for a click to start the camera on a mobile phone and the application processes the information contained in the QR code. The program decodes the information and then processes it in the background. After receiving the information, it tries to connect to the IP address located in the QR code.

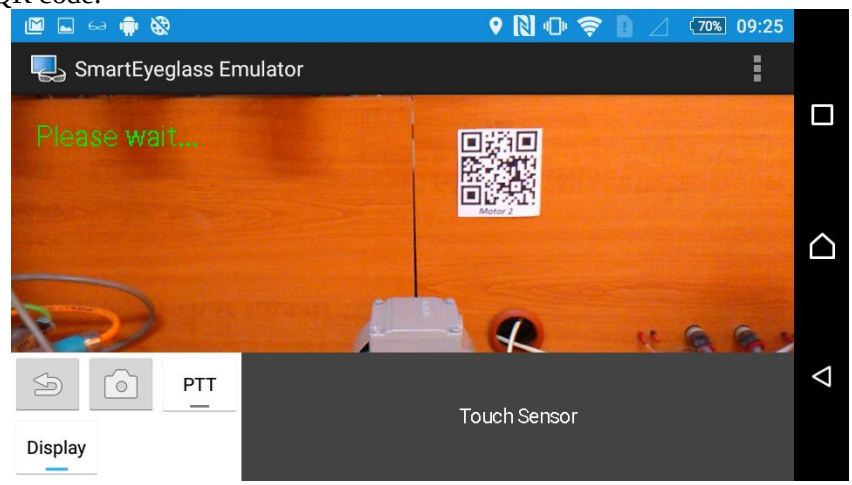

Fig. 5 The QR code decoding process

If the PLC is available, the "PLC AVAILABLE" message is displayed, followed by the values that are automatically loaded and displayed on the display for a specific engine. If an application fails to connect to the PLC, it will print an error message.

C. Engine control

After successful connection to the PLC, a main screen is displayed, which contains the data on the observed, controlled engine. The principle of control lies in the touch-sensitive surface of the emulator called the Touch Sensor. When the first shift is to the right, the engine is turned on. With the next shift shifted, its speed increases and the engine shaft starts to move clockwise (right). When moving the sensor to the left, the speed will be reduced. After the value 0 is exceeded and the left-hand drive is moved continuously, the motor changes the clockwise counterclockwise direction (left).

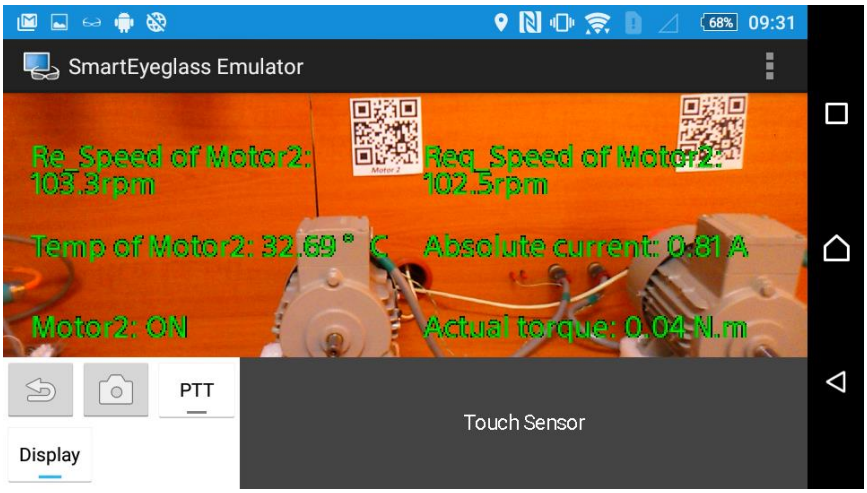

Figure 6 Main application view

The main screen of the entire application displays the values of the monitored engine described in chap. 2.5 It contains the following values:

- Re_Speed_Motor1 – real speed of engine
- Req_Speed_Motor1 – requested engine speed
- Temp of Motor1 – engine temperature
- Absolute current– current flowing through the engine
- Motor1 – ON: turned ON, OFF: turned OFF
- Actual torque – actual torque of engine

Keeping the touch sensor longer (LONGPRESS), the application returns to the start screen and allows us to track another engine. The original engine turns off at this step and its speed is set to 0 for safety.

D. Hardware and software application requirements

The proposed software includes the requirements that the smartphone that the app installs must meet. Smartphone Requirements:
• Supported Android,
• Applications needed to run the proposed software,
• needed free smartphone memory,
• Memory RAM size.
A prerequisite for the smartphone is that it must contain the version of Android min. 4.4 (KitKat), or must support miniaturized updates to this version.
Apps required for the proper functioning of the software:
• Smart Connect - a basic application that includes all Sony devices and connects apps across devices or between multiple devices. It is a necessary part because the emulator is designed as a real device. Without this app, the smartphone does not have access to the emulator.
• SmartEyeglass - the main application that controls the SmartEyeglass emulator in the background.

• SmartEyeglass Emulator - Simulator Reflecting the Principle of Real Smart Glasses. It serves as the graphical interface of SmartEyeglass.
• S7_SSG - The proposed application itself. S7 means that it is data reading and control of Siemens PLC. SSG stands for: Sony SmartEyeglass.
Each application occupies a certain amount of storage as well as RAM. The table shows how much memory the individual applications need to run the entire software. The SmartEyeglass emulator does not use RAM because it only serves as the visual interface of SmartEyeglass.

Table 2
Requirements for smartphone memory

		RAM memory [MB]	
Application	**Phone memory [MB]**	**average**	**max.**
Smart Connect	5,78	4,3	34
SmarEyeglass	18,64	14	24
SmartEyeglass Emulator	2,82	-	-
S7_SSG	5,62	16	29

V. CONCLUSION

The main purpose was to design and implement software for smart glasses containing Android OS and their connection to one of the most important Siemens PLC automation components. The software should be designed for physical smart glasses. However, this goal was not met due to the availability of smart glasses in Slovakia. However, the emulator is available to demo the functionality. The application should be implemented over several years at Magneti Marelli s.r.o Slovakia to monitor production process data. Siemens is still the largest manufacturer and distributor for the European automation and management market. For this type of PLC, a library is available that is capable of linking the Android OS with a PLC to ensure communication and sharing of information.

REFERENCES

[1] Glassappsource.com: Differences between smart glasses [online]. [cit. 2018-03-15]. Available on the internet : <http://www.glassappsource.com/specs/1/14,8,7,1/epson-moverio-bt200-specs-vs-vuzix-m100-specs-vs-recon-jet-specs-vs-google-glass-specs>.

[2] Siemens AG: S7:300 CPUs [online]. [cit. 2017-04-11]. Available on the internet: <http://w3.siemens.com/mcms/programmable-logic-controller/en/advanced-controller/s7-300/cpu/Pages/Default.aspx >.

[3] Siemens AG: Sinamics 110 Basic Servo Drivers [online]. [cit. 2017-04-11]. Available on the internet: <https://w3.siemens.com/mcms/mc-solutions/en/converters/low-voltage-converters/sinamics-s/positioning-motor/pages/sinamics-s110.aspx>

[4] Sourceforge.net: Moka7 Project overview [online]. [cit. 2018-04-15]. Available on the internet: <http://snap7.sourceforge.net/moka7.html>.

[5] Difference Between Synchornous and Induction Motor [online]. [cit. 2018-04-17]. Available on the internet https://www.electricaleasy.com/2015/06/difference-between-synchronous-and-induction-motor.html.

[6] *Smart glasses technology and applications*, [online]. [cit. 2017-04-18]. Available on the internet *http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.643.9521&rep=rep1&type=pdf*

[7] Czerniak, J. M., Zarzycki, H., Apiecionek, Ł., Palczewski, W., & Kardasz, P. (2018). A Cellular Automata-Based Simulation Tool for Real Fire Accident Prevention. Mathematical Problems in Engineering, 2018

[8] CZERNIAK, Jacek M.; ZARZYCKI, Hubert. Artificial acari optimization as a new strategy for global optimization of multimodal functions. Journal of computational science, 2017, 22: 209-227