FACTA UNIVERSITATIS
Series: **Mechanical Engineering** Vol. 10, N° 1, 2012, pp. 71 - 80

WIRELESS DEVICE CONTROL VIA USB PORT [*]

UDC 681.5

Laslo Tarjan, Ivana Šenk, Stevan Stankovski[#], Gordana Ostojić

University of Novi Sad, Faculty of Technical Sciences, Serbia
[#]E-mail: stevan@uns.ac.rs

Abstract. *Control of peripheral devices directly from a personal computer is a possibility that lowers automation costs at the locations where computers are already being used for other purposes. All new-generation computers come with built-in USB ports which can accommodate an interface providing for required digital inputs and outputs for the control of peripherals. Wireless technology enables dislocation of the executive part of the interface, while the computer remains in control without the necessity to trace cables from the computer to the controlled device. This paper proposes an interface which enables wireless 'on/off' control of peripheral devices via USB computer port. The control algorithm is executed directly on the computer, by a dedicated application, which communicates with the executive module of the interface via USB port and a wireless link. The interface and the computer can communicate both ways, which enables the user to easily implement his/her own control application if required.*

Key words: *Wireless Communication, Device Control, USB*

1. INTRODUCTION

Control of peripheral devices directly from a PC computer is very handy in cases when a user interface must be started up, when a small number of inputs/outputs has to be controlled with a relatively small reliability. The possibility of input and output control is also usable in applications where such peripherals are used that have to be connected to a computer, e.g. applications with RFID readers, such as identification at parking lots [1], identification of products in production systems [2,3], etc. As regards the control system reliability, this is not an optimal solution, since a PC computer can 'freeze' at any moment (due to user fault, or OS error, etc.), thus leaving the controlled periphery without control. Such variant is seldom a reality, but it should be taken into consideration. However, there

Received October 17, 2011

[*] **Acknowledgment**. This paper is a result of the project financed by the Ministry of Education, Science and Technological development of Republic of Serbia (Project No. 35001: *Automated System for Identification and Object Tracking in Industrial and Nonindustrial Automation*).

are some applications where a brief absence of control can be tolerated, or is unnoticed, and in such cases direct control from a PC computer is a viable option.

In the twentieth century, when an LPT port was used for connecting peripheral devices, such as mouses, printers, scanners, etc., this port could also be used for a simple control of digital inputs. The I/O pins of LPT port can directly be programmed by the user [4]. Newer generations of computers are delivered without LPT and COM ports (RS232), and instead are equipped with USB ports. The reason for this switch is the universality of USB, which is present on all modern peripheral devices (mouse, keyboard, printer, scanner, etc.).

Basically, USB communication as a serial one. For the purpose of controlling peripheral devices via digital inputs/outputs, it is necessary to allow buffering of wanted I/O states as a single bit data for every I/O. The devices which come with built-in USB ports, already have some form of USB controller. However, to communicate with the peripherals which exclusively require digital I/Os, an additional interface must be included as a repeater between the USB port and digital I/Os.

Bearing in mind that industrial environment does not always allow the computer to be placed beside the machine which needs to be controlled, it is necessary to solve the problem of connecting the control interface, on the one side, and the controlled device, on the other. The simplest solution of this problem is to use wireless connection between the control interface and the PC.

2. Proposed Solution

For solving the problem of controlling peripheral devices directly via USB port, a modification of an already published modification of a microcontroller-based intermediate interface is suggested [5]. According to the proposed solution, radio links should be added to allow the module for connection with the USB port, and the executive module of the interface, to be separated, and thus allow the connection of digital inputs and outputs of the controlled peripheral device. The wireless link would provide greater flexibility of operation. Block diagrams of the principal solution for the proposed interface are shown as follows:

- communication module (Fig. 1),
- executive module (Fig. 2).

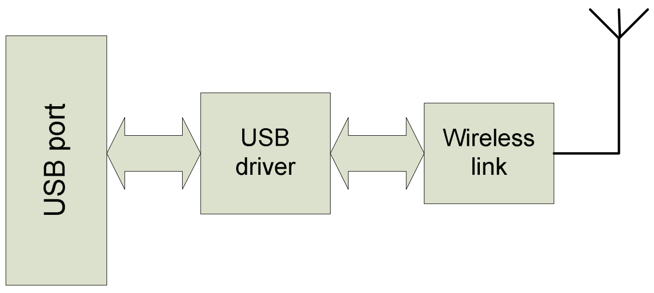


Fig. 1 Block diagram of the communication module

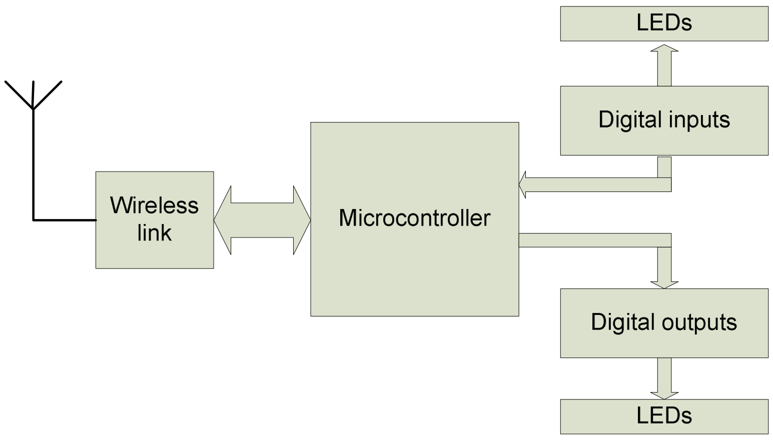


Fig. 2 Block diagram of the executive module with digital I/Os

The communication module has to redirect the incoming commands from the control application to the USB port of a computer, to the executive module, which is remote by a couple of hundreds of meters. Operation at a minimum distance of 300 meters, outdoors, or up to 50 meters indoors, should be provided.

In order to protect the microcontroller which controls the I/O signals, galvanic separation of input pins from the microcontroller input should be provided, due to differences in voltages used for logical levels. In addition, it should be possible to accept the digital signals with a wider range of voltages for logical levels.

Powerful outputs are also required, which allow direct on/off switching of various actuators, such as: relays, spools of electro-pneumatic distributors, smaller electromotors, etc., or the control components of actuators.

LEDs are required for signaling the I/O state, which light up when there is a logical unit connected to I/O.

Communication module should be powered via the USB port, within the USB power limits, while the executive module should be powered from the available power source on the controlled peripheral device. This facilitates the interface exploitation by the user, since no additional power source is required.

2.1 Choice of technology

Following components are selected for building the interface between the USB port and the peripheral device controlled via the digital I/Os:

- microcontroller ATmega8
- radio module XBee PRO
- integrated circuit FT232RL
- driver circuit ULN2803
- optocouplers PC847

as well as some passive auxiliary components, which are required for reliable device operation.

The interface consists of two parts:

- Communication module: USB↔XBee
- Executive module: XBee↔Controller↔U/I

Executive module is the principal component of the interface (Fig. 2a). Central part of the executive module is ATmega8 microcontroller [6] which controls the values on output pins based on the instructions from the PC. On the other side, the controller sends the states of its digital inputs to the computer. Input pins are scanned according to a preset time interval, which, in this case, equals 5 ms. This time interval can be changed from the user interface (min. value is 2 ms, max. 500 ms). This setting is allowed from the control application on the PC, while the instructions are sent in a serial mode.

The digital inputs on the USB interface are realized using the ULN2803 driver circuits [7]. They use the NPN outputs, which, in the state of logical one, connect the output pin to the ground (GND), while in the state of logical zero the output remains disconnected, i.e., it can be connected to the power source (max. 50 V) via a resistor. Each output pin can conduct 500 mA current. This limitation is acceptable since this current is sufficient for controlling the relay spool, or electro-pneumatic distributor, electromagnet, electric lock, etc.

The digital inputs are galvanically separated from the input microcontroller pins by PC847 optocouplers [8]. In this way, the interface is protected from interference. The galvanic separation allows the use of voltage levels higher than the TTL levels, such as the microcontroller input, since the signals do not arrive directly at microcontroller input pins, but are transferred via the optoelement. Such approach provides 0 to 30V input ranges, for the digital input pins of the interface.

In order to facilitate visual monitoring of I/O state, each pin is connected to a yellow LED. The LED lights up when the I/O state is logical one, or is otherwise inactive. This method of I/O state identification is very important since it allows an elegant monitoring of control interface operation. For lower power consumption, low-power (3 mA) LEDs are used.

Communication between the executive and communication modules goes via XBee PRO radio links [9]. These radio links utilize 802.15.4 or ZigBee communication standard, depending on their intended application, and operate on a 2,4 GHz frequency. It is recommended not to use the ZigBee standard unless a *mash* topology is required, which is why the modules are configured according to 802.15.4 standard. XBee radio modules can be hooked up to a wireless network, while each module has its own unique, factory set IP address. There are two variants of these modules, the basic XBee, and the more powerful XBee-PRO. Table 1 gives comparison of XBee and XBee-PRO module characteristics (Digi International, 2008).

The communication between the executive module microcontroller, and the USB port on the computer, is established via the communication module (Fig. 3). The section of communication module which establishes communication with the USB port, is realized using FT232RL integrated circuit, by FTDI. This circuit is in fact a USB-to-RS232 converter, which comes with Windows drivers. The drivers can be downloaded from the manufacturer's web page [10]. Once the driver is installed, the PC reports a virtual COM port, and the ongoing communication runs as if through a hardware COM port (RS232). Wireless communication from the output of FT232RL circuit to executive module's microcontroller, is allowed via XBee PRO radio links. Communication pins of FT232RL circuit are TTL compatible, and can be directly connected to the communication pins of the XBee PRO radio link. Since the XBee communication pins operate with 0-3,3V voltage, it is necessary to use the parallel connected Zener diodes, with a 3,3V threshold. XBee PRO modules with external antenna provide ranges of 75-80 m indoors, while the outdoor range goes up to 800 m [11].

Table 1. Comparison of XBee modules' characteristics

	XBee	XBee-PRO
Power voltage	2,8 – 3,4 V	2,8 – 3,4 V
Transmit current	45 mA (at 3,3 V)	150 mA (at 3,3 V)
Receive current	50 mA (at 3,3 V)	55 mA (at 3,3 V)
Range - indoors	do 30 m	do 60 m
Range - outdoors	do 90 m	do 750 m
Power	1 mW	10 mW
Data rate (RF)	250,000 b/s	250,000 b/s
Serial data rate	1200 b/s – 250 kb/s	1200 b/s – 250 kb/s
Receiver sensitivity	-92 dBm	-100 dBm
Operating frequency	2,4 GHz	2,4 GHz
Operating temperature	-40 – 85 ºC	-40 – 85 ºC
Dimensions	243,8 x 276,1 mm	243,8 x 329,4 mm

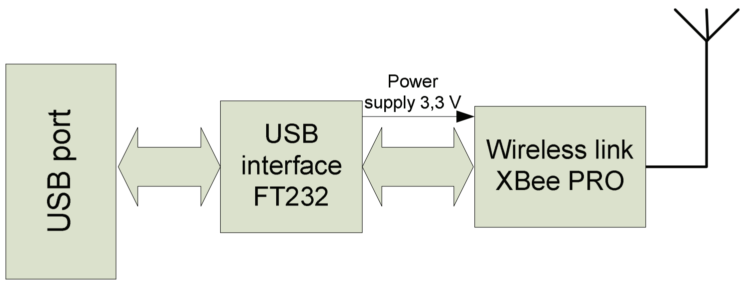


Fig. 3 Block diagram of communication module

On the other side, the executive module also features a XBee PRO radio link, which establishes communication with the communication module, thus forming a two-way data exchange, from the USB port to digital outputs, and from the digital inputs to the USB port, i.e., to the computer which runs the control software application. Communication pins on XBee module are directly connected to microcontroller's *Rx* and *Tx* pins, protected by 3,3V Zener diodes. This module requires a constant 3,3V power source, since the XBee modules require that power voltage. To keep the integrated circuit of the executive module as simple as possible, the microcontroller is also powered by 3,3V, thus voltage levels of logical signals range from 0V to 3,3V. The output stage is powered directly from the industrial power grid, by 5-30V DC. The regulated 3,3 V voltage level is provided by L78L33 integrated circuit [12]. This circuit is internally protected from power and temperature surges, which renders it practically indestructible. With adequate cooling it can deliver up to 100 mA of current. The input voltage range is 5 to 30 V. Shown in Fig. 4 is the executive module block diagram.

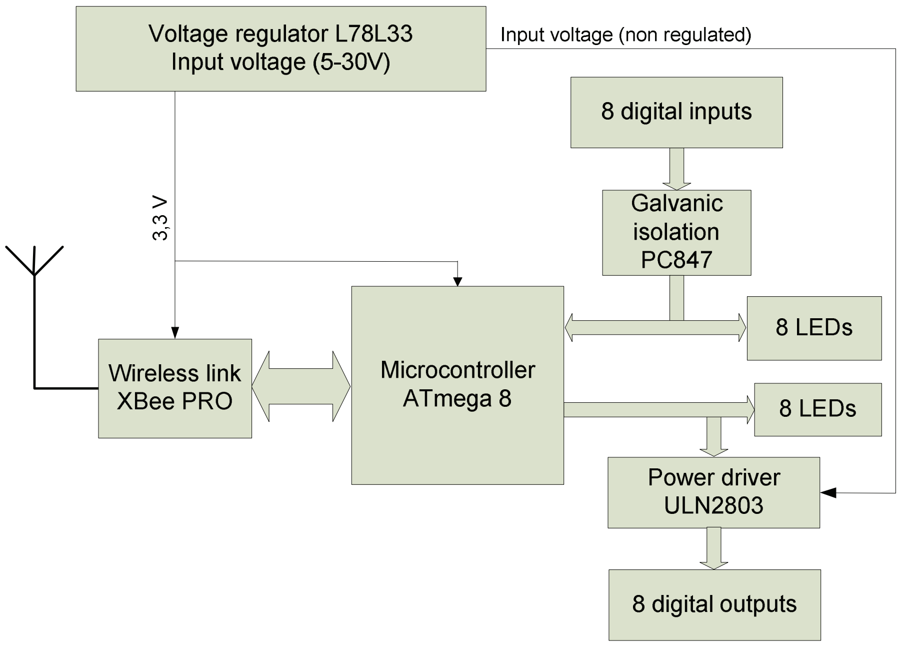


Fig. 4 Block diagram of executive module

Communication module is powered directly from the USB port, and requires no additional external power sources. The USB 2.0 compliant port can provide up to 500 mA of constant current [13]. For USB port protection, the power unit of the interface is equipped with a 4.7 mH damper, which prevents fluctuations of current, and a 500 mA fuse, which protects the port from overload damage.

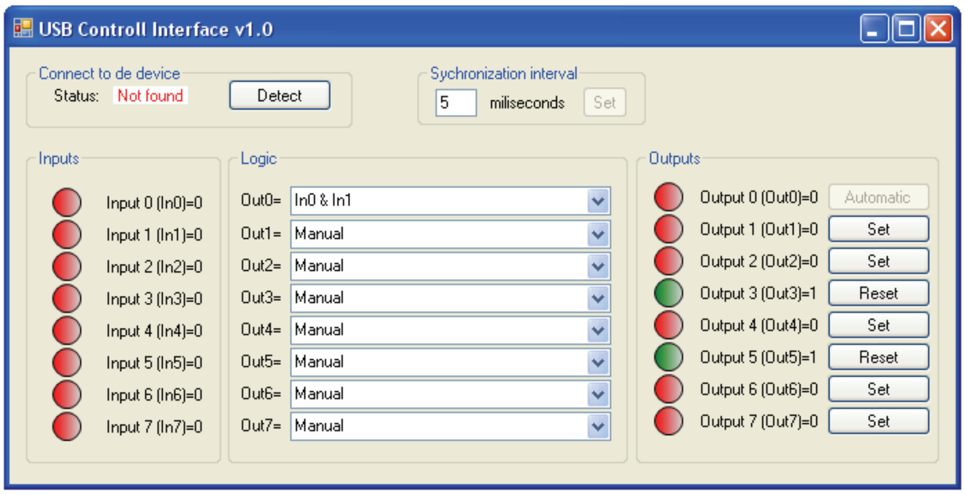
3. User Interface on PC Computer

The main control component is the PC computer. To allow direct control from the computer using the developed wireless interface, a Windows software application is designed using *Visual Basic*-u *9.0* (Fig. 5).

The main window of the software application is divided in two parts (Fig. 5). The upper part contains controls for interface setup. By pressing the *Connect* button, the software application recognizes which of the virtual COM ports is connected to the interface. The status window shows current status of the connection. There are three different states: device is not found (*Not found*), communication with device is established (*Connected*), and communication with device is broken (*Disconnected*). Once the connection with the

USB control interface is established, the 'Connected' status is indicated. In cases of communication disruptions which last longer than 1 s, the status changes to 'Disconnected'. In this case the software tries to re-establish the connection. If the connection is successfully established, the status again changes to 'Connected'.

One of the adjustable communication parameters is the time interval required for synchronization of states of digital inputs/outputs. When the software is run, the default synchronization time interval is set at 5 ms, until changed by the user. Minimum allowed time interval is 2 ms, while the maximum is 500 ms.


USB Controll Interface v1.0

Connect to de device
Status: Not found
Detect

Sychronization interval
5 miliseconds
Set

Inputs
Input 0 (In0)=0
Input 1 (In1)=0
Input 2 (In2)=0
Input 3 (In3)=0
Input 4 (In4)=0
Input 5 (In5)=0
Input 6 (In6)=0
Input 7 (In7)=0

Logic
Out0= In0 & In1
Out1= Manual
Out2= Manual
Out3= Manual
Out4= Manual
Out5= Manual
Out6= Manual
Out7= Manual

Outputs
Output 0 (Out0)=0 Automatic
Output 1 (Out1)=0 Set
Output 2 (Out2)=0 Set
Output 3 (Out3)=1 Reset
Output 4 (Out4)=0 Set
Output 5 (Out5)=1 Reset
Output 6 (Out6)=0 Set
Output 7 (Out7)=0 Set

Fig. 5 USB control user interface

Shown in the other part of the window are the states of digital inputs (Fig. 5, left), digital outputs (Fig. 5, right), and a dropdown list with logic expressions which form the output states based on input states (Fig. 5, middle). The state of input/output is indicated by a small circle which can be red (logical zero is on the input/output pin) or green (logical one is on the input/output pin), and is labeled by the text which describes the state.

For each output pin it is possible to define the way in which the output state is formed. If manual mode is selected, the output state is formed by clicking the appropriate button. If some of the available logic formulas for output forming is selected, the button for output state change becomes inactive, while the output state is formed based on the selected formula and the states of particular inputs.

In the initial version of this software, only the basic logic operations between inputs, AND, and OR, have been implemented. Should some other type of control be required, the source code can be easily extended with a new logic expression. Here, of course, some knowledge of *Visual Basic* is required, to modify the source code as well as to re-compile it.

3.1 Communication between interface and software application

The mode of communication between the USB control interface, and software application is established to allow simple transfer of the state of input to the computer, as well as the state of output from the computer to interface.

The software application sends a three-byte instruction from the computer, where the first byte represents device address, the second represents the wanted state of output, and the third byte is a control one. The control byte is formed as the sum of the previous two bytes, taking only the lower eight bytes. A sent word has following form:

<ADDRESS> + *<OUTPUT>* + *<CONTROL>*

The software application expects to receive a four-byte answer to the instruction it has sent. The first byte represents the device address, the second byte is formed based on the state of input, while the third byte is formed based on the real state of the output. The control byte is again formed in the manner previously described. The instruction that comes from the USB port takes following form:

<ADDRESS> + *<INPUTS>* +*<OUTPUTS>* + *<CONTROL>*

By forming answers to sent instructions in this manner, the software application that runs on PC always keeps record of whether or not the interface has received and executed the sent instruction. In this way, feedback loop between the USB control interface, and the control software application is constantly maintained.

Serial communication is performed at a rate of 38400 bits per second. Bearing in mind that it is a simple RS232 communication, the USB control interface can communicate with any other software application as well.

4. Conclusion

The paper presents a way of controlling peripheral devices via digital input/output, directly from a computer. With high computing power being commonly available nowadays, with at least several unused USB ports, it would be a waste not to make use of these resources. An adequate intermediate interface allows direct control of low-consumption peripheral devices, via digital input/output. Both the wireless communication between the communication module attached to the computer and the interface executive module provide for great flexibility in application of this hardware.

The paper describes the principle realization of a USB microcontroller-based control interface, which uses an integrated circuit to convert USB serial communication into RS232 serial communication. It also utilizes XBee PRO modules which enable communication between the communication and the executive device modules. The realized interface can control peripheral devices via 8 digital inputs, and 8 digital outputs. The distance between the computer and the controlled device can be up to 80 m indoors, or even up to 800 m outdoors. Specified in this paper are the constituent interface modules, while the dedicated Visual Basic application is also described. Also provided is a detailed description of the mode of communication between the interface and the computer-run controlled application.

The device proved stable during testing, with no significant problems in functioning.

As regards further research, the proposed concept should be enhanced by adding other executive modules, which would increase the number of controlled devices as well as the operating range. The XBee modules configured in *mash* mode act as repeaters, which provides an opportunity to automate the entire environment in this way, using just a single communication module and the required number of executive modules. Furthermore, this interface could be extended with a couple of analogue inputs and outputs, to allow for analogue sensors' connectivity and actuators controlled by some analogue parameter. In this way, control of a number of different peripheral devices would be enabled.

Another possibility for improvement is the addition of an RS232 serial port onto the interface itself, to allow for a wireless connection to the host computer of peripherals which require RS232 connection. Since modern computers come without serial ports, this enhancement would represent a solid advantage.

In the case of using a larger number of executive modules, the computer-run control application would require modifications. In order to obtain a more functional control application, the existing number of logic expressions should be extended to allow better control of outputs based on input states. Finally, a function which enables the input/output states to be recorded and stored in a file, would also represent a significant improvement.

REFERENCES

1. Pala Z., Inanc N., 2009, *Utilizing RFID for smart parking applications*, Facta Universitatis, Series Mechanical Engineering, University of Nis, Vol.7, No 1, pp. 101 – 118.
2. Stankovski S., Ostojić G., Jovanović V., Stevanov B., 2006, *Using RFID technology in collaborative design of the assembly systems*, Facta Universitatis, Series Mechanical Engineering, University of Nis, Vol.4, No 1, pp. 75 – 82.
3. Purić R., Lazarević M., Ostojić G., Marović B., Stankovski S., 2011, *RFID-based lifecycle product monitoring for insurance purposes*, Facta Universitatis, Series Mechanical Engineering, University of Nis, Vol.9, No 1, pp. 119-126.
4. Axelson J., 1996, Parallel port complete: Programming, Interfacing & Using the PC's parallel Printer Port, Lakeview Reasearch.
5. Tarjan L., Šenk I., Ostojić G., 2010, *Control of peripheral devices using an USB port (in serbian)*, 9. Infoteh-Jahorina, Elektrotehnički fakultet Istočno Sarajevo, Sarajevo, Bosina and Hercegovina, Vol. 9, Ref. E-II-13, pp. 550-553.
6. Specification for microcontroller ATmega 8, available at www.datasheetcatalog.org/ datasheet/atmel/2486S.pdf (accessed October 2011).
7. Specification for power driver ULN2803, available at www.datasheetcatalog.org/ datasheets/90/366828_DS.pdf (accessed October 2011).
8. Specification for optocoupler PC847, available at www.datasheetcatalog.org/ datasheets/90/36063_DS.pdf (accessed October 2011).
9. Digi International, 2008, *XBee/XBee-PRO OEM RF Modules Product Manual*, available at http://ftp1.digi.com/support/documentation/90000982_A.pdf (accessed October 2011).
10. FTD International Ltd., *Specification datasheet for FT232RL*, available at www.ftdichip.com/Products/FT232R.htm (accessed October 2011).
11. Ignjatović I., Tarjan L., Dudić S., Šešlija D., 2010, *Testing range wireless system for monitoring the state of filter cartridges in pneumatic systems (in serbian*), 9. Infoteh-Jahorina, Elektrotehnički fakultet Istočno Sarajevo, Sarajevo, Bosina and Hercegovina, Vol. 9, Ref. C-4, pp. 317-320.
12. Specification for voltage regulator L78L33, available at www.datasheetarchive.com/pdf-datasheets/Datasheets-25/DSA-480726.html (accessed October 2011).
13. Specification for Universal Serial Bus 2.0, available at www.usb.org/ developers/docs/usb_20.zip\usb_20.pdf (accessed October 2011).

UPRAVLJANJE PERIFERIJAMA BEŽIČNIM PUTEM PREKO USB PORTA

Laslo Tarjan, Ivana Šenk, Stevan Stankovski, Gordana Ostojić

Upravljanje periferijskim uređajima direktno preko PC računara je jedna od mogućnosti, koja snižava troškove automatizacije, na mestima na kojima se svakako već koristi računar u druge svrhe. Svaki računar novije generacije poseduje USB port, na koji se može priključiti potreban interfejs, koji bi obezbedio potrebne digitalne ulazne i izlazne signale za upravljanje periferijama. Bežična tehnologija omogućava izmeštanje samog izvršnog dela interfejsa, a da se upravljanje i dalje vrši sa računara, i da se pri tome ne moraju provlačiti kablovi od računara do uređaja kojim se upravlja. Tema ovog rada je prikaz interfejsa koje omogućava "on/off" upravljanje perifernim uređajima preko USB porta računara bežičnim putem. Algoritam upravljanja se izvršava direktno na računaru, pomoću namenski pisane aplikacije, koja preko USB porta i bežičnog linka komunicira sa izvršnim delom interfejsa. Komunikacija između interfejsa i računara je dvosmerna i realizovana je na taj način da korisnik lako može napisati svoju aplikaciju za upravljanje ukoliko ima potrebe za tim.

Ključne reči: *bežična komunikacija, upravljanje periferijama, USB*