Organizing the Synchronous Remote Labworks in Institutions of Technical Engineering Education

Bohdan Sus[a], Oleksandr Bauzha[a], Sergiy Zagorodnyuk[a], Valentyna Maliarenko[a]

[a] *Taras Shevchenko National University of Kyiv, 64/13 Volodymyrska Street, City of Kyiv, 01601, Ukraine*

Abstract

Implementation of the synchronous remote labwork system has been discussed in details. Original approach to controlling an arbitrary laboratory devices has been presented. Effective operational interaction system between the teacher and students and complete control over the quality and completeness has been provided. A structure and components of remote labwork that includes microcontroller development board, broadcast imaging device, assistant microcontroller with 4-pole relay have been suggested. A workstation configuration that allow any number of students to remotely connect and operate on the computer was was explained. The mode of coordination of students actions has been discussed. It is shown that in the ideal case, when the program does not need to be corrected and reworked, the student spends 90% of the time on developing use the working time of computers and in particular the operating time of the specified devices as rationally as possible. Program structure of the additional microcontroller Chipkit that allows the student operate with real laboratory hardware in the mode of synchronous remote laboratory work has been presented.

Keywords

microcontroller, synchronous remote labwork, broadcast imaging device

1. Introduction

The instantaneous spread of the infectious disease COVID-19 worldwide has fundamentally changed almost all spheres of human life. The education system belongs to the most social spheres of life, so it is not surprising that it had to be changed first. To prevent the severe consequences of the COVID-19 spread, educational institutions are forced to implement various electronic education mechanisms [1]. The mechanisms include synchronous network platforms for collective interaction [2, 3] as well as asynchronous distance learning technologies [4]. In addition to the globally spread disease of COVID-19, other important reasons for the introduction of electronic education technologies also remain relevant for many regions of the planet. In particular, for Ukraine, such a reason is open aggression on the part of despotic totalitarian empires.

The birth and development of distance education began even before the spread of the infectious disease COVID-19. As a result of the evolution and interconnection of such systems, a range of universal learning management systems (LMS) [5], integrated learning management systems (IMS) [6], and environment-specific management systems [7]. Such systems are implemented to develop, manage and distribute educational materials.

Indeed, for teaching lectures, modern LMS systems allow easy use of network and video technologies [8]. Instead conducting laboratory work with complex physical, engineering, or chemical equipment involves performing specific operations directly with this equipment, and performing each of these operations in a remote mode needs an individual solution [9, 10]. There are different ways of

ICST-2023: Information Control Systems & Technologies, September 21-23, 2023, Odesa, Ukraine
EMAIL: bnsuse@gmail.com (B. B. Sus); asb@univ.kiev.ua (Oleksandr. S. Bauzha); szagorodniuk@gmail.com (S. P. Zagorodnyuk); VMalyarenko12@gmail.com (V. M. Maliarenko)
ORCID: 0000-0002-2566-5530 (B. B. Sus); 0000-0002-4920-0631 (Oleksandr. S. Bauzha); 0000-0003-3415-7746 (S. P. Zagorodnyuk); 0000-0003-4585-3114 (V. M. Maliarenko)

organizing laboratory work in such difficult conditions. The most common of these are virtual labs [11], remote labs [12, 13] and intelligent expert systems [14].

Virtual laboratories, computer modeling, and simulation play an important role in such diverse scientific fields as physics, meteorology, neurology, nanoscience, sociology, economics, and archaeology [15]. The importance of models is not limited to scientific research, simulators of various complex experimental installations of vehicles, mechanisms, and special equipment have also become widespread [16]. In particular, the training of aircraft pilots begins with simulation devices. Students can gain the experience necessary for their further education as a result of performing a cycle of virtual experiments and laboratory work [11]. In the case of the implementation of laboratory work in the form of virtual experiments [17], simulation cannot completely substitute a real experiment, since the simulation is valid only within the defined range of phenomena, the studied process belongs to [18]. If you do not go beyond this range of phenomena, then the virtual experiment will fully satisfy the requirements and criteria of the experiment [19].

Among the variety of innovative approaches in education, the gamification of learning is widely used, which contributes to increasing the interest and involvement of students in learning. Works [20, 21] consider several laboratory works and classes that contribute to a deeper motivation of students to study and involve students in research work.

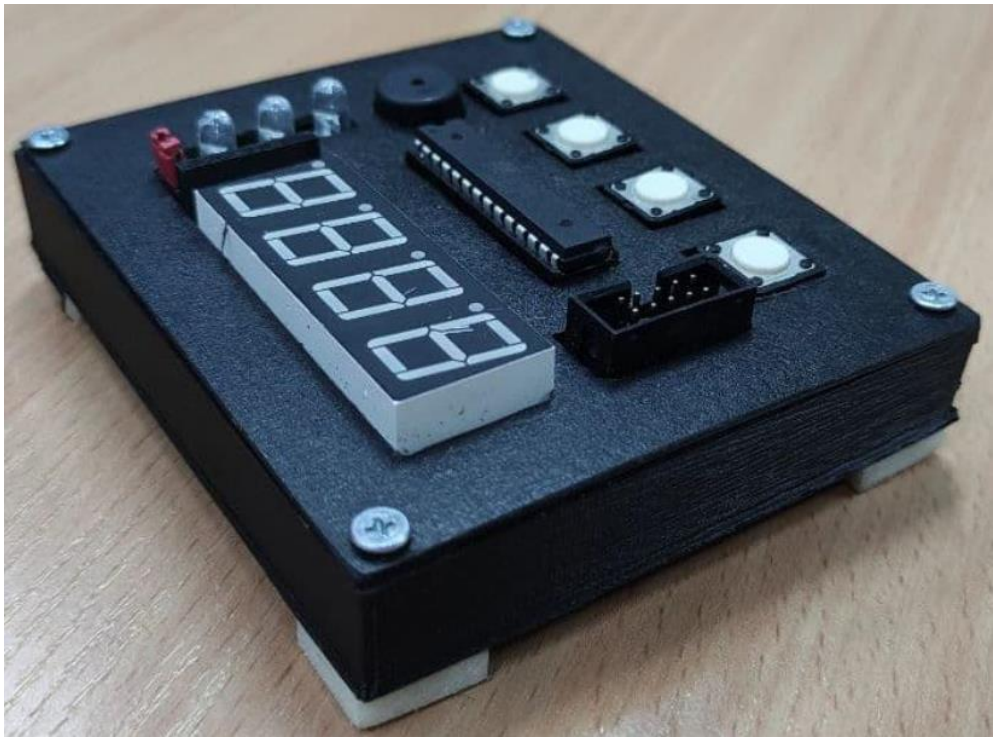

Figure 1: Development board with the microcontroller Atmel ATMega8.

Many scientific works consider the task of remote control of an experiment [22], which allow students to perform practical and laboratory work similar to those carried out in physical laboratories [23, 24].

The set of considered approaches eloquently testifies to the importance of organizing a remote mode of conducting laboratory work and performing individual experimental research. Such a remote mode cannot now be considered as an optional additional accessory that only increases the convenience of the educational and research process. On the contrary, in many cases, it is the only possible mechanism for carrying out laboratory work or an experiment. Therefore, the development and improvement of technics for remote control of laboratory equipment remains quite relevant and fully corresponding to world trends now [25].

In recent decades, remote laboratories have been widely implemented in engineering education processes and integrated into e-learning frameworks offered to engineer and natural science students.

Remote laboratory activities are used to support the life cycle of scheduled synchronous learning, as well as the organization of autonomous student learning. The document [26] provides an overview of modern technologies in the development of remote laboratory work and a description of the most vivid and relevant examples of the implementation of these technologies. In this article, the authors offer a practical and reliable implementation example of the synchronous remote labwork. This study [27] is based on the wide application of popular and common general-purpose microcontrollers in education process. They clearly demonstrate and emphasize the need to consider remote laboratories as methodical pedagogical material, which is a significant and inseparable part of any synchronous remote labwork. A new and original approach to controlling an arbitrary laboratory device is presented, which includes a limited number of buttons for manual input of information and an indicator for visual output of information. The main goal of the work is to ensure effective operational interaction of the teacher with students and complete control over the quality and completeness of the laboratory assignment.

2. Design of the synchronous remote labwork

The learning management system can be easely replicated since it consists of only popular and globally common components. In particular, in the educational laboratory where the authors teaching, 7 copies of such a learning management system have already been created. System has been successfully and stably operated for 1.5 years. Each LMS instance runs independently of other instances of the same LMS.

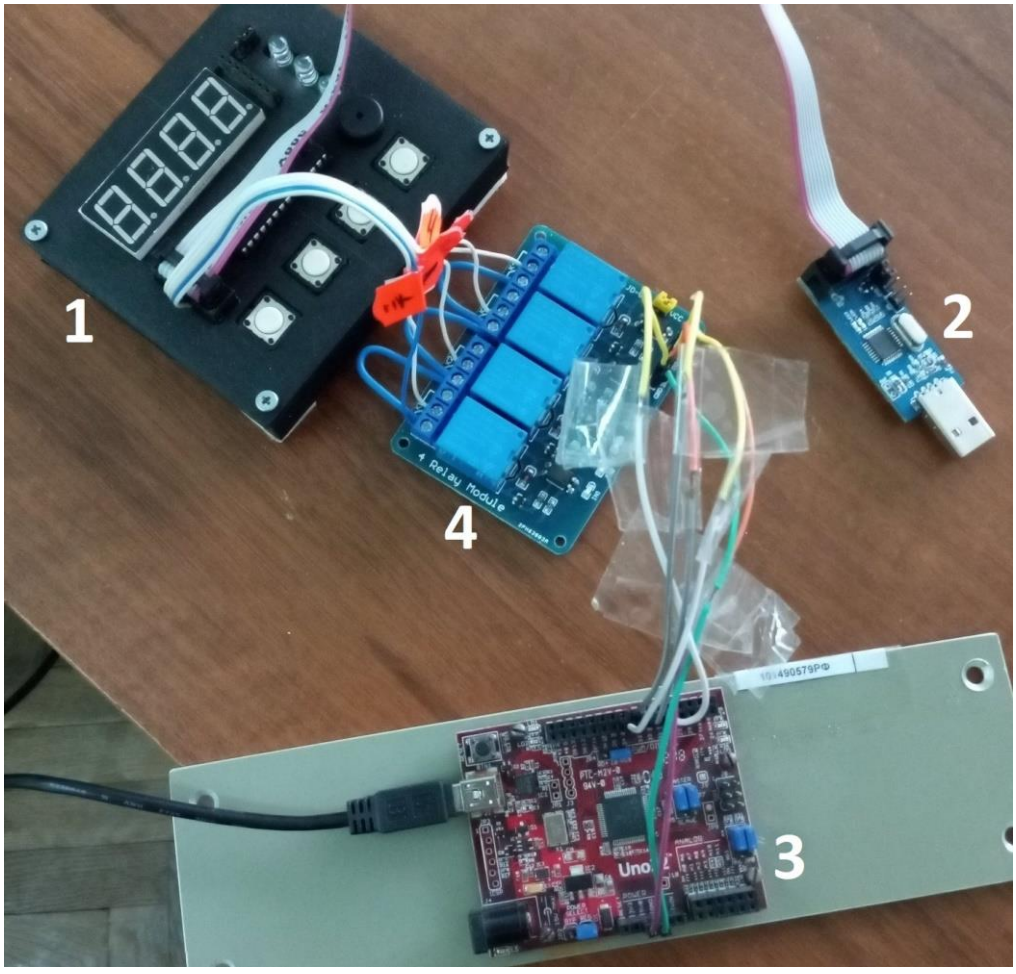

Figure 2: Development board with the microcontroller Atmel ATMega8.

As an example presented in the article, the object of research for students is the programming of Atmel ATMega8 microcontrollers, and the object of study is a development board with this controller. The layout board (fig. 1) contains a matrix with an ATMega8 controller in the form of a DIP microcircuit; digital screen with four seven-segment digital indicators; a loudspeaker that allows you to output a sound signal; four buttons that can be pressed simultaneously or in any sequence. The board is connected by a 10-pin cable to a USBasp programmer. The USBasp programmer, in turn, is connected to the USB port of a workstation or laptop.

Previously, before the introduction of quarantine restrictions and the declaration of martial law, students worked with such mock-up boards directly: they programmed the controller through the USBasp programmer, connecting it to personal laptops, pressing buttons, and checking the result of the program by reading information on the screen. Unfortunately, due to the reasons listed above, students cannot connect breadboards to their laptops and work directly with development boards now. Therefore, to fully replace such traditional work with mock-up boards, a laboratory learning management system (LMS) was created, consisting of the following hardware components:

1. A development board with an Atmel ATMega8 microcontroller and a USBasp programmer;

2. A webcam with a USB interface, which is defined in the system as a source of live broadcast (Imaging Device) and transmit a picture for the computer with a small pixel image from 480×320 to 720×480.

3. Microcontroller ChipKit Uno32, to which a 4-pole low-level relay is connected (fig. 2). This means that the relay connected to a logical zero when the control signal is applied. Since the four buttons of the breadboard have one common contact, the breadboard is connected to the executive contacts of the 4-pole relay by means of a 5-pin loop. As a result, the ChipKit microcontroller plays the role of the student's "remote hand", with which he can independently press the buttons of the breadboard in any combination and in any sequence. Previously, before the introduction of quarantine restrictions and the declaration of martial law, students worked with such mock-up boards directly: they programmed the controller through the USBasp programmer, connecting it to personal laptops, pressing buttons, and checking the result of the program by reading information on the screen. Unfortunately, due to the reasons listed above, students cannot connect development boards to their laptops and work directly with development boards now. Therefore, to fully replace such traditional work with development boards, a laboratory learning management system (LMS) was created, consisting of the following hardware components:

4. A workstation with the MS Windows Server operating system, which can be configured as a terminal server with RDP (Remote Desktop Protocol) access, will allow any number of students to remotely connect and operate on the computer.

All of the listed components make up one instance of a learning management system (LMS). A webcam directed at the board allows student to observe its current image in real-time. A workstation or laptop must have three free USB ports: the first for the USBasp programmer, the second for the ChipKit microcontroller, and the third for the web camera. As a result, an authorized student located in any country in the world can connect to this workstation, write, debug and write a program for the ATMega8 microcontroller to the breadboard, run it for execution, press the buttons of the breadboard with the help of the ChipKit microcontroller and through the camera look at the seven-digit indicators.

3. Design of the synchronous remote labwork

To start work, the student receives the IP address and TCP port necessary to establish a remote RDP session, the login and password combination from the teacher. The login must be unique within one operating system installed on the workstation. For example, the student receives the following access parameters from the teacher:

IP socket.....91.202.129.107:42161
Login.....Student1
Password.....4535365270

and opens a remote RDP session from his home computer. It can be done, for example, by other students who log on to the same computer identified by an IP socket with logins Student2 and Student3 and the corresponding passwords. But then students need to understand very clearly which operations they can do simultaneously, in parallel, and which, on the contrary, strictly sequentially, one after the other. The number of students simultaneously working on the same operating system depends on what part of laboratory work students can do simultaneously and independently, and what part only in the mode of coordination of their actions.

For example, the authors teach laboratory work in the discipline "Microprocessor engineering" and use seven computers with the MS Windows Server operating system for this purpose. AVR Studio 5.0 integrated programming environment installed on each operating system, in which students create a new project for the target ATMega8 microcontroller and write a program in the C programming language. Experience shows that an average student writes a program in about 30-60 minutes, but it is possible to test the operation of the compiled program only on the microcontroller. So, a student spends 3-6 minutes on programming the ATMega8 microcontroller and checking the operation of the program on the microcontroller. As you can see, in the ideal case, when the program does not need to be corrected and reworked, the student spends 90% of the time on developing the program, not using devices connected to the computer's USB bus at all. Obviously, many students can do this part of the work on one computer at the same time. Accordingly, the student spends 10% of the time checking the program and using USB devices - a camera, a programmer, and a ChipKit microcontroller for pressing buttons, and during the use of these three devices by one student, each of them must be blocked from accidental or intentional use by another student. Therefore, when more than one student works on the same computer, several students can finish creating and compiling the program the same time and proceed to programming the ATMega8 microcontroller. So the question arises "who is first?" and a queue may form.

So, the authors faced two tasks. First of all, use the working time of computers and in particular the operating time of the specified USB devices as rationally as possible. Secondly, toprevent the formation of a long queue of students who have to wait for the release of USB devices to check the operation of their program. In order to perform these two tasks simultaneusly, it is advisable to allocate one computer for two students, the first of which enters and works in the Windows Server operating system under the login Student1, the second - under the login Student2. Thus, 14 students can simultaneously perform laboratory work on 7 computers:

Computer	**External IP socket**	**Internal IP socket**	**User logins**
1	91.202.129.107:42161	10.0.22.101:3389	Student1, Student2
2	91.202.129.107:42162	10.0.22.102:3389	Student1, Student2
3	91.202.129.107:42163	10.0.22.103:3389	Student1, Student2
4	91.202.129.107:42164	10.0.22.104:3389	Student1, Student2
5	91.202.129.107:42165	10.0.22.105:3389	Student1, Student2
6	91.202.129.107:42166	10.0.22.106:3389	Student1, Student2
7	91.202.129.107:42167	10.0.22.107:3389	Student1, Student2

Each user's desktop has a shortcut to launch only the AVR Studio IDE, so each of the two students can launch independently of the other. Instead, there are no shortcuts for using USB devices on the desktop. programmer program is used to program the eXtreme Burner program is used to push the buttons of the breadboard with the ChipKit microcontroller PuTTYis used to view the webcam VLC Media Player. A student cannot launch all three programs using a shortcut. Instead, the authors developed a special program loader of these three programs under the conventional name Loader (fig. 3).

The software has 4 buttons. The first three of them launch eXtreme Burner, PuTTY, and VLC Media Player. But it is impossible to predict the result of pressing each of these buttons. This result depends on whether another student has run the same program. If this program has already been launched by another student. When you click on the button, a message will appear that the program has been launched by another student. Otherwise, pressing the button successfully launches the program. The fourth button sends an interactive message to all users of the system, which

immediately appears on the desktop of each user, including the author of the message on top of all programs (fig. 3). The message is distributed by the system command:

msg * Please free up the USB devices!

Another student working in a pair with the author of the message immediately sees this interactive dialog box and can do different things: send another message in response, close all programs to test the operation of the program on the ATMega8 microcontroller, perform both of these actions, or, conversely, ignore the message. Therefore, students have a reliable instant channel of communication between themselves, which allows them to use their time as efficiently as possible and achieve positive result. The full text of the **Loader** program in C# is show on fig. 4.

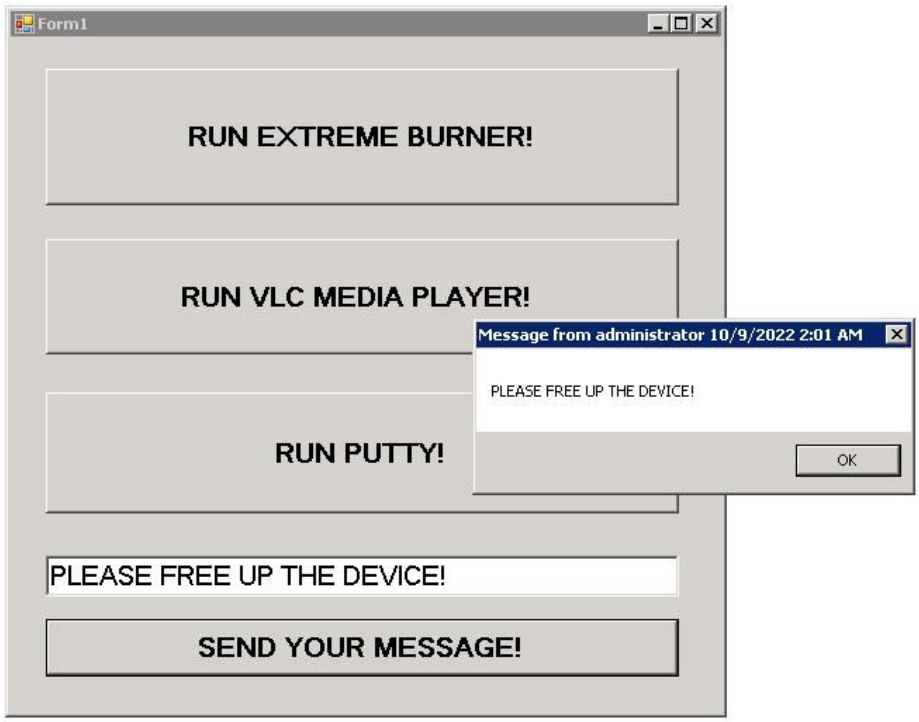

Figure 3: Interface of developed program "Loader".

It is easy to see that when the program starts, only one statement InitializeComponents();, and all the rest of the program code consists of 4 functions that handle the events of pressing each of the 4 buttons.

For a brief summary, only the **B1_Click()** and **B4_Click()** event handlers are given in the article, because the **B2_Click()** and **B3_Click()** event handlers are identical to the **B1_Click()** event handler, but in the function **cm1.Parameters.Add("ProcessName", "AVRProg");** instead of "**AVRProg**" have values "**vlc**" and "**putty**", as well as in the function **Process.Start("C:\\Users\\Burner.lnk");** have the corresponding full path to the program launch.

4. Program Structure Of The Chipkit Uno32 Microcontroller

Microcontroller interacts with the operating system via the USB bus. Due to FT232RL microchip of FTDI on this microcontroller, the operating system emulates a serial port of the standard UART interface with a bitrate of 9600 bit/s. In the MS Windows operating system, such a serial port is called COM13, and in the GNU Linux operating system - TTY13. Instead of the number 13, there can be another number in the range 3-17.

Any terminal program can be used to interact with devices via standard UART/RS232/RS485 interfaces: HyperTerminal, MiniCOM, PuTTY. In synchronous remote laboratory work, the popular PuTTY is used, the name of which expresses for itself (Put–>TTY).

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Diagnostics;
using System.Management.Automation;
using System.Management.Automation.Runspaces;
using System.Collections.ObjectModel;
namespace Loader
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }
      private void button1_Click(object sender, EventArgs e)
      {
         string A=String.Empty;
         int i = 0;
         RunspaceConfiguration runspaceConfiguration = RunspaceConfiguration.Create();
         using (Runspace RS = RunspaceFactory.CreateRunspace(runspaceConfiguration))
         {
            RS.Open();
            Pipeline pipeline = RS.CreatePipeline();
            Command cm1 = new Command("Get-Process");
            cm1.Parameters.Add("ProcessName", "AVRProg");
            Collection<PSObject> cr = null;
            using (Pipeline pl1 = RS.CreatePipeline())
            {
               pl1.Commands.Add(cm1);
               cr = pl1.Invoke();
            }
            foreach (PSObject R in cr)
            {
               i++;
            }
         }
         if (i == 0)
         {
            Process.Start("C:\\Users\\Burner.lnk");
         }
         else
         {
            MessageBox.Show("Одна копія процесу AVRProg вже запущена!!!");
         }
      }
      private void button4_Click(object sender, EventArgs e)
      {
         Process P = new Process();
         P.StartInfo.FileName = "C:\\Windows\\system32\\msg.exe";
         P.StartInfo.Arguments = " * " + textBox1.Text;
         P.Start();
      }
   }
}
```

Figure 4: C# code of developed program "Loader".

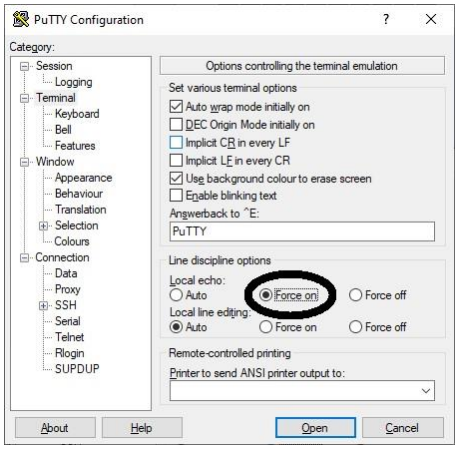

Figure 5: C# code of developed program "Loader".

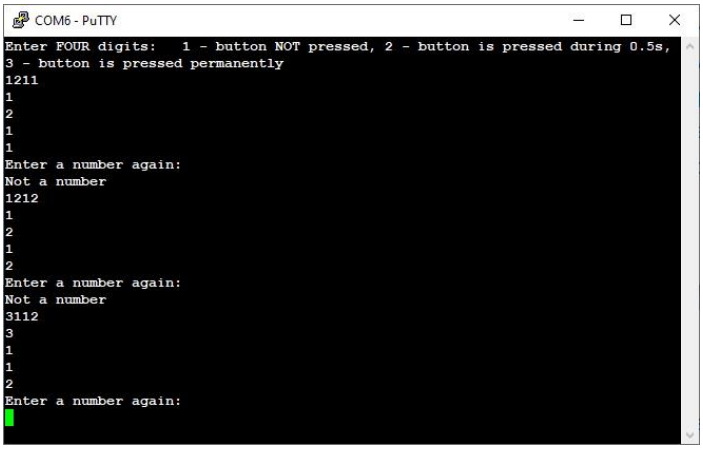

Figure 6: PuTTY work session on microcontroller ChipKit UNO32.

PuTTY must be configured as a serial TELNET client to a ChipKit microcontroller configured as a TELNET server. In this way, the student can type any commands on the keyboard as a sequence of characters, and when the ENTER button is pressed, these commands should be executed, just like on a real TELNET server accessible over a TCP/IP network. To do this, it is necessary to change a parameter of the PuTTY program, which is incompatible with working in TELNET mode. When the user of the PuTTY program creates a new connection session using the SERIAL protocol, by default the characters entered from the keyboard are sent only to the interface channel and are not duplicated on the screen of the PuTTY program itself. As a result, the PuTTY user cannot see what command he entered and whether he made a mistake when entering it. For the entered keys to be sent to the interface channel and displayed on the screen at the same time, it is necessary to change the value of

the **Terminal|Local echo** from the **AUTO** position to the **FORCE ON** (fig. 5). As a result, echo works - the entered command is simultaneously displayed on the screen and sent to the device.

When the user successfully establishes a connection, a short instruction on the screen appears: "Enter FOUR digits: 1 - button NOT pressed, 2 - button is pressed during 0.5s, 3 - button is pressed permanently" (fig. 6). It follows that the user can enter a command consisting of 4 digits from 1 to 3, for example 3121, 1211, 2113. Any other command that does not correspond to this format will be ignored. For that, the buttons of the breadboard with the ATMega8 microcontroller are numbered 1-4, and 1 is the leftmost button, 4 is the rightmost button (fig. 6). Therefore, in order to press and release the button number 1, and not to press the other buttons, user needs to execute the command 2111, that is, the number 2 must be in the first position in the command. Similarly, commands 1211, 1121, and 1112 press and release the second, third, and fourth buttons. The 2222 command sequentially presses and after 0.5s releases all four buttons in sequence, but at any given time only one or none of the buttons is pressed.

```
char rx_byte = 0;
String rx_str = "";
boolean not_number = false;
int T,b4,b3,b2,b1;

int P=13; int P1=31; int P2=32; int P3=33; int P4=34;
void setup() {
  pinMode(P,OUTPUT);
  pinMode(P1,OUTPUT);   pinMode(P2,OUTPUT);
  pinMode(P3,OUTPUT);   pinMode(P4,OUTPUT);
  Serial.begin(9600);
  Serial.println("Enter FOUR digits:  1 - button NOT pressed...");
  D1(); D2(); D3(); D4();
}
void loop() {
  if (Serial.available() > 0) {    // is a character available?
    rx_byte = Serial.read();       // get the character
    if ((rx_byte >= '1') && (rx_byte <= '3')) {
      rx_str += rx_byte;
    }
    else if (rx_byte == '\r') {
      if (not_number) {
        Serial.println("Not a number");
      }
      else {
        digitalWrite(P,HIGH);
        T=rx_str.toInt();
        b4=T%10;
        b3=((T-b4)/10)%10;
        b2=((T-10*b3-b4)/100)%10;
        b1=((T-100*b2-10*b3-b4)/1000)%10;
        // print the result
        //Serial.begin(9600);
        Serial.println(T);
        Serial.println(b1);
        switch (b1) {
          case 1:  D1(); break;
          case 2:  S1(); break;
          default: R1();
        }  delay(1000);
        Serial.println(b2);
        switch (b2) {
          case 1:  D2(); break;
          case 2:  S2(); break;
          default: R2();
        }  delay(1000);
        Serial.println(b3);
        switch (b3) {
          case 1:  D3(); break;
          case 2:  S3(); break;
          default: R3();
        }  delay(1000);
        Serial.println(b4);
        switch (b4) {
          case 1:  D4(); break;
          case 2:  S4(); break;
          default: R4();
        }  delay(1000);
        Serial.println("Enter a number again:");
        digitalWrite(P,LOW);
      }
      not_number = false; rx_str = ""; //clear string for reuse
    }
    else {
      not_number = true;    // flag a non-number
    } } }
void R1() {  digitalWrite(P1,HIGH);  }
void D1() {  digitalWrite(P1,LOW);   }
void S1() {  R1(); delay(100); D1(); }
void R2() {  digitalWrite(P2,HIGH);  }
void D2() {  digitalWrite(P2,LOW);   }
void S2() {  R2(); delay(100); D2(); }
void R3() {  digitalWrite(P3,HIGH);  }
void D3() {  digitalWrite(P3,LOW);   }
void S3() {  R3(); delay(100); D3(); }
void R4() {  digitalWrite(P4,HIGH);  }
void D4() {  digitalWrite(P4,LOW);   }
void S4() {  R4(); delay(100); D4(); }
```

Figure 7: C code of program for microcontroller ChipKit UNO32.

Instead, to control a program running on an ATMega8 microcontroller, it may be necessary for buttons with specific numbers to be pressed exactly at the same time for a short time. It is easy to do, but for this user needs to execute not one, but two commands in sequence. Example 1: in order to simultaneously press button 1 and 4, two commands must be executed: 3112, 1111. Example 2: in order to simultaneously press all buttons 1,2,3,4, two commands must be executed: 3332, 1111.

Command 1111 is required to release all buttons on the breadboard and prepare it to press another key or key combination. Any correct command is executed exactly for 4 seconds.

To interpret such commands consisting of 4 digits, a special program is composed on the ChipKit controller (fig. 7). The program consists of two parts, the first of which is executed once when the controller is turned on or restarted. The second part is the body of an infinite loop that is executed when any button on the keyboard is pressed.

In the first part, all variables involved in the program are declared and their initial values are assigned. Pins **P**,**P1**,**P2**,**P3**,**P4** are configured to output information: pin **P** is connected to the microcontroller's built-in LED, which lights up for 4 seconds when a command is executed, **P1-P4** assigned to the control contacts of relay, each pole of which pushes the corresponding button of the breadboard, all contacts are assigned the value of logical zero with the help of statements **D1(); D2(); D3(); D4();**. These four statements complete the first part of the program.

The second part of the program begins with the construction "**void loop() {**". If the entered character turned out to be a number from 1 to 3. Then it is appended to the current line of the string type: **rx_str += rx_byte;**. If the entered character is not a number from 1 to 3, it is checked whether this character is the ENTER key. If true, the command execution procedure is started. If not true, the message "Not number" is displayed to the user (fig. 6) and he can enter a new command without an error.

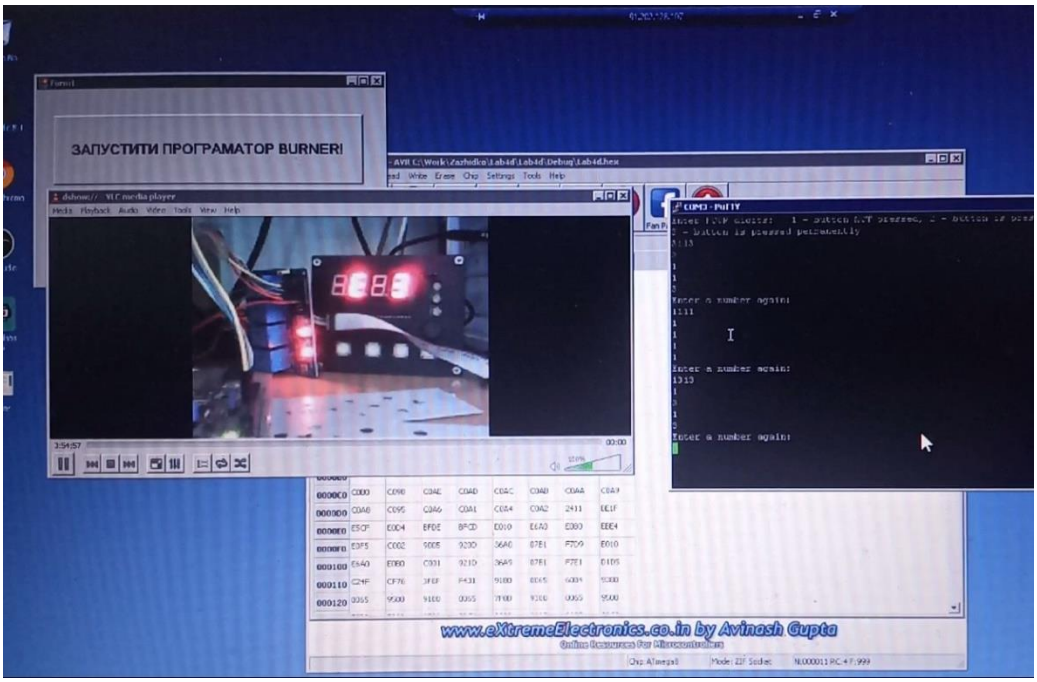

Figure 8: Screenshot of student computer when he is demonstrating developed lab project working on microcontroller ATMega8.

The procedure for executing a correctly entered command is as follows. The LED with contact P lights up, signaling the execution of the command: **digitalWrite(P,HIGH);**. The accumulated string of 4 digits from 1 to 3 is converted into a number of type Int: **T=rx_str.toInt();**. The following operators calculate the entered numbers **b1**,**b2**,**b3**,**b4**. Next, the analysis of the first number **b1** is performed using the **switch(b1)** operator, and depending on its value at pin **P1**, the value of logical zero or one is set. Similarly, other numbers **b2-b4** are sequentially analyzed and values are set to contacts **P2-P4**. At the end of the cycle, the prompt "**Enter a number again:**" is displayed to the user, the LED with the contact turns off, which visually means the end of the command execution: **digitalWrite(P,LOW)**, the line with the accumulated digits of the current command is cleared: **rx_str = "";**, this completes the execution of the command. The user gets the opportunity to enter and execute the next command, this cycle is repeated.

As a result, the student's working environment consists the five programs: the AVR Studio environment, the eXtreme Burner programmer, the PuTTY terminal program, the VLC Media Player and the Loader program (fig. 8). In combination with network platforms for collective interaction, a student is able to consult with teachers and other students, demonstrate his screen and report on the completed laboratory work.

5. Outcome discussion

The authors have spent significant time to find similar hardware and software solutions for remote control of real laboratory equipment and to compare the parameters of these solutions with the built complex described in this work. But it was not possible to find at least one published similar solution that would be a complete or partial analogue to the solution created by the authors of the article.

Great progress has been made by the Hungarian IT company LabShare.com (https://TheLabShare.com), but the solution of this company is exclusively a software complex that allows one researcher to measure any amount of laboratory data of a biological, genetic, sociological or technical nature on his own in a traditional non-remote way, then distribute this data to a cloud network resource. The same solution allows other researchers or young student researchers to process, sort, filter or otherwise use accumulated and stored data, including building neural networks and artificial intelligence systems.

There are many other software products that allow students to remotely configure various server applications, including databases, web servers, web services, file servers, and development and execution environments. Again, all of these solutions are purely software solutions.

Finally, there are many server programs that offer students to control a virtual software emulator, which replaces the functionality of an equipment. There are many publications on this topic, but it is fair to note that such situation, when a real laboratory device was successfully used by students for several years in the laboratory, and then the students continued to use the same device remotely using the Internet, was practically not considered by engineers.

Therefore the remote control system for a breadboard with an ATMega8 microcontroller built and described in the article is a truly new and universal system that allows you to remotely press an arbitrary number of buttons on any laboratory instrument or set of instruments and with the help of a camera visually in direct mode ether to observe the behavior of such devices. The instrument buttons can be pressed simultaneously or sequentially.

On many devices, the buttons have one common contact, so the "remote student hand" can be schematically implemented using a 4-pole or 8-pole relay. On the contrary, when the buttons of the device do not have common contacts, instead of multi-pole relays, separate single-pole relays must be used, but the C program for the ChipKit UNO32 microcontroller (fig. 7) does not change from this, because it works on a different "software" abstraction level.

6. Prospects of system development

The system of remote control of laboratory equipment described in the article has already been used for 3 years in conducting laboratory work in the specialized engineering disciplines "Microprocessor Technology" in the specialty "Computer Engineering" and "Electronic Physics" in the specialty "Applied Physics". Both specialties are studied at the Taras Shevchenko Kyiv National University (https://rex.knu.ua/en).

At this time, the implementation of the remote equipment control system in the laboratory workshop on the discipline "Oscillations and waves" as a section of general physics, where it is necessary to remotely control spectrum analyzers, frequency meters, oscilloscopes and phase meters is ongoing. In the future, it is also planned to use the same system to perform laboratory work from another section of general physics "Electricity and magnetism", as well as from the engineering discipline "Radio circuits and signals". In both laboratory workshops, it is planned to use a system for commutation of electromagnets and current frames, switching on and off fragments of electric circuits, selection the values of passive elements including capacitors, resistors, inductors, oscillating circuits, semiconductor devices.

7. Conclusions

Modern information technologies allow to build a specialized network channel, which allows authorized users to provide reliable and stable access to the computers of the internal private network of the educational institution and all programs running on these computers. An almost unlimited number of students can work remotely on the same computer. If one copy of a specific program is installed on one computer, but several students remotely log in to this computer, the program can be downloaded to the desktop of each student.

If students must use programs that use external devices connected to the operating system via USB/FireWire/RS232 interfaces, only one user should be able to run such a program. To organize the queue of students, an additional loader must be used that starts the corresponding program only after checking the same program process is not launched by another student.

Laboratory computers on which students work remotely can have an additional microcontroller or a microcomputer to which an executive device is connected, for example, a multipole contactor. Such a device can help the student remotely push buttons and change the position of the switches of the laboratory equipment. Images of the hardware itself can be observed by students using a webcam opened in streaming media players such as VLC Media Player.

References

[1] M. Zhenchenko, O. Melnyk, Y. Prykhoda, Z. I., Experience of use of electronic educational resources by ukrainian teachers during the distance learning due to the covid-19 pandemic, CEUR Workshop Proceedings 3104 (2021) 55–65. URL: http://ceur-ws.org/Vol-3104/paper158.pdf.

[2] S. Jacques, A. Ouahabi, T. Lequeu, Synchronous e-learning in higher education during the COVID-19 pandemic, in: 2021 IEEE Global Engineering Education Conference (EDUCON), IEEE, 2021, pp. 1102–1109. doi:10.1109/EDUCON46332.2021.9453887.

[3] D. Turnbull, R. Chugh, J. Luck, Learning Management Systems and Synchronous Communication Tools, 1 ed., Routledge, 2021, pp. 39–49. doi:10.4324/9781003125921-5.

[4] V. Lytvyn, O. Akimova, H. Kuznetsova, T. Zenchenko, O. Stepanenko, I. Koreneva, The use of synchronous and asynchronous teaching methods in pedagogical education in covid-19 terms, International Journal of Health Sciences, 5 (2021) 617–629.

[5] Lms selection guide 2020 (2021). URL: https://learn.trakstar.com/resource-center?postType=literature.

[6] Project 1edtech: teaching & learning innovation, 2020 (2020). URL: https://www.1edtech.org.

[7] A. Ak, V. Topuz, A. Altıkardeş, B. Oral, Development of a remote laboratory infrastructure and LMS for mechatronics distance education, EURASIA Journal of Mathematics, Science and Technology Education, 14 (2018). doi:10.29333/ejmste/89947.

[8] J. Guerra, O. Maria, Multimedia as an efficient web-based tool for the development of communicative competence of students studying english for specific purporses, World Journal on Educational Technology, 6 (2014) 273–277.

[9] A. K. Mohammed, H. M. El Zoghby, M. M. Elmesalawy, Remote controlled laboratory experiments for engineering education in the post-COVID-19 era: Concept and example, in: 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES), IEEE, 2020, pp. 629–634. doi:10.1109/NILES50944.2020.9257888.

[10] M. Uğur, K. Savaş, H. Erdal, An internet-based real-time remote automatic control laboratory for control education, Procedia - Social and Behavioral Sciences 2 (2010) 5271–5275. doi:10.1016/j.sbspro.2010.03.859.

[11] T. Chaikivskyi, O. Bauzha, B. Sus, N. Tmienova, S. Zagorodnyuk, 3d simulation of virtual laboratory on electron microscopy, in: N. Kryvinska, I. Izonin, M. Gregus, A. Poniszewska-Maranda, I. Dronyuk (Eds.), CEUR Workshop Proceedings, volume 2533, CEUR-WS, 2019, pp. 282–291.

[12] A. Cardoso, V. Sousa, P. Gil, Demonstration of a remote control laboratory to support teaching in control engineering subjects, IFAC-PapersOnLine, 49 (2016) 226–229. doi:10.1016/j.ifacol.2016.07.181.
[13] C. Monzo, G. Cobo, J. A. Morán, E. Santamaría, D. García-Solórzano, Remote laboratory for online engineering education: The RLAB-UOC-FPGA case study, Electronics, 10 (2021) 1072. doi:10.3390/electronics10091072.
[14] B. Sus, S. Zagorodnyuk, O. Bauzha, V. Maliarenko, T. Zahorodniuk, Development of practical exercises in educational institutions using intelligent expert systems, in: 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T), IEEE, 2021, pp. 279–284. doi:10.1109/PICST54195.2021.9772242.
[15] D. Reid, J. Burridge, D. Lowe, T. Drysdale, Open-source remote laboratory experiments for controls engineering education, International Journal of Mechanical Engineering Education, 50 (2022) 828–848. doi:10.1177/03064190221081451.
[16] T. Knuuttila, M. Merz, E. Mattila, Computer models and simulations in scientific practice, Science & Technology Studies, 19 (2021) 3–11. doi:10.23987/sts.55199.
[17] B. Sus, I. Revenchuk, N. Tmienova, O. Bauzha, T. Chaikivskyi, Software system for virtual laboratory works, in: 2020 IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT 2020 - Proceedings, volume 1, Institute of Electrical and Electronics Engineers Inc., 2013, pp. 396–399. doi:10.1109/CSIT49958.2020.9322046.
[18] E. Arnold, J. Kästner, When can a computer simulation act as substitute for an experiment? a case-study from chemisty, 2013. URL: http://philsci-archive.pitt.edu/12970/.
[19] B. Sus, I. Revenchuk, O. Bauzha, S. Zagorodnyuk, Virtual laboratory as custom e-learning implementation and design solution, in: CEUR Workshop Proceedings, volume 2833, CEUR-WS, 2013, pp. 177–187.
[20] J. Głowacki, Y. Kriukova, N. Avshenyuk, Gamification in higher education: Experience of Poland and Ukraine, Advanced Education, 5 (2018) 105–110. doi:10.20535/2410-8286.151143.
[21] B. Sus, N. Tmienova, I. Revenchuk, O. Bauzha, S. Stirenko, Gamification approach to the creation of virtual laboratory works and educational courses, in: CEUR Workshop Proceedings, volume 2711, CEUR-WS, 2020, pp. 68–78.
[22] I. Mundilarto, H. D. Surjono, Development of light polarization experimental apparatus for remote laboratory in physics education, Physics Education, 56 (2020) 015008. doi:10.1088/1361-6552/abc4da.
[23] T. Caetano, M. F. Rezende Junior, A. Pina da Silva, C. C. Moreira, The physics remote laboratory: implementation of an experiment on standing waves, European Journal of Physics, 43 (2020) 025801. doi:10.1088/1361-6404/ac4978.
[24] C. Gravier, N. Abdellaoui, Adaptive follow-up of online engineering laboratories activities, International Journal of Online and Biomedical Engineering (iJOE), 6 (2020) 32. doi:10.3991/ijoe.v6i3.1336.
[25] Z. Laouina, L. Ouchaouka, A. Elkebch, M. Moussetad, M. Radid, Y. Khazri, A. Asabri, Manufacturing and developing remote labs in physics for practical experiments in the university, in: M. E. Auer, D. May (Eds.), Cross Reality and Data Science in Engineering, volume 1231, Springer International Publishing, 2021, pp. 193–204. doi:10.1007/978-3-030-52575-0_16, series Title: Advances in Intelligent Systems and Computing.
[26] L. Gomes, S. Bogosyan, Current trends in remote laboratories, IEEE Transactions on Industrial Electronics, 56 (2009) 4744–4756. doi:10.1109/TIE.2009.2033293.
[27] W. A. Salah, B. A. Zneid, Evolution of microcontroller-based remote monitoring system applications, International Journal of Electrical and Computer Engineering (IJECE), 9 (2019) 2354. doi:10.11591/ ijece.v9i4.pp2354-2364.