Web server with ATMEGA 2560 microcontroller

Article *in* IOP Conference Series Materials Science and Engineering · February 2016

DOI: 10.1088/1757-899X/106/1/012018

8 authors, including:

Eugen Raduca
Universitatea "Eftimie Murgu" Reşiţa
51 PUBLICATIONS **136** CITATIONS
SEE PROFILE

Cornel Hatiegan
Babeş-Bolyai University
86 PUBLICATIONS **355** CITATIONS
SEE PROFILE

Silviu Draghici
Universitatea "Eftimie Murgu" Reşiţa
32 PUBLICATIONS **77** CITATIONS
SEE PROFILE

Cristian Paul Chioncel
Babeş-Bolyai University
99 PUBLICATIONS **360** CITATIONS
SEE PROFILE

Web server with ATMEGA 2560 microcontroller

2016 IOP Conf. Ser.: Mater. Sci. Eng. 106 012018

(http://iopscience.iop.org/1757-899X/106/1/012018)

Web server with ATMEGA 2560 microcontroller

E Răduca[1], D Ungureanu-Anghel[2], L Nistor[1], C Hațiegan[1], S Drăghici[2], C Chioncel[1], E Spunei[1] and R Lolea[1]

[1]Eftimie Murgu University of Resita, Department of Electrical Engineering and Industrial Informatics, P-ta Traian Vuia, no. 1-4, 320085 Resita, Romania
[2]Politehnica University of Timisoara, Department of Automatic and Applied Sciences, Vasile Parvan str., no. 1-2, 300223 Timisoara, Romania

E-mail: e.raduca@uem.ro

Abstract. This paper presents the design and building of a Web Server to command, control and monitor at a distance lots of industrial or personal equipments and/or sensors. The server works based on a personal software. The software can be written by users and can work with many types of operating system. The authors were realized the Web server based on two platforms, an UC board and a network board. The source code was written in "open source" language Arduino 1.0.5.

1. Introduction

With the spread of the Internet, computer networks have been continuously developed and in this context and network servers aimed mainly communication network clients. Current servers are able to provide solutions and information services increasingly various users (customer network) and the main reason for choosing to study and build a Web server, is to obtain a device that is capable of offering both services and information to command, control and monitoring other devices that require this.

One of the main objectives of the work, is the data communication network, whether local or the Internet, with equipments a [1], [2] and/or sensors [3] or at a distance [4]. This communication will be indirect, because between the equipment to be controlled and user is interleaved Web server based on microcontroller. The protocol used for this server in this project is the HTTP protocol. Language for building web pages is HTML that allows viewing pages with most browsers and thus permits communication with Windows, Linux, Android etc. The server is designed to enable users not having spot devices like PC, Tablet, Smartphone etc. be able to control and monitor remote the server peripherals connected to the devices it manages.

2. Design Server

2.1. Figures Block diagram of the server

The server is accomplished on base of two platforms, one [5] built around ATMEGA 2560 microcontroller [6] and other [7] based W5100 network controller [8]. Network controller interface is provided of W5100 circuit manufactured by WIZnet and command and control network controller will be in the load ATMEGA 2560 microcontroller, product in AVR architecture of Atmel. Interconnecting the two platforms, having regard to the objectives pursued was performed according to Figure 1.

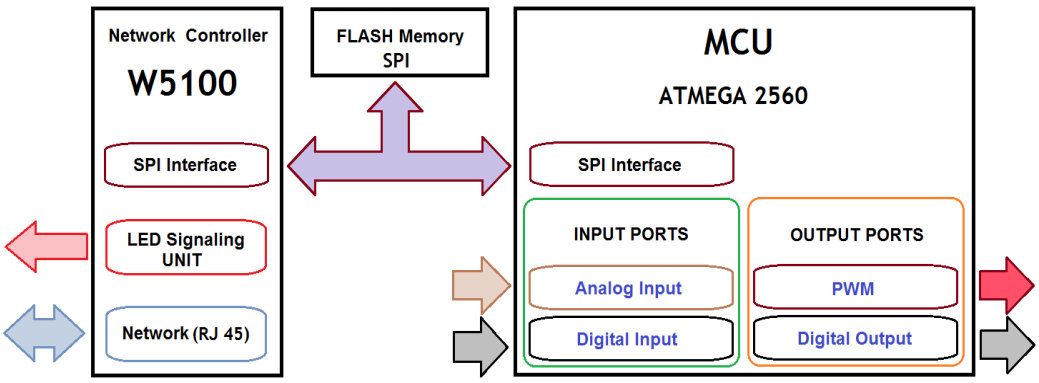

Figure 1. Block diagram of ATMEGA 2560 Web Server

Making server was designed in order to use it as a command entity, control and remote monitoring through networks, of some equipment from industry and personal privat. The user who handles the said equipment has the function of client and, from network platform for own internet, connects from anywhere to the Web server ATMEGA 2560 with cable UTP and RJ45 connector. The Web server, through its communication components with the outside, input and output ports are connected from the UC platform [5], to the interest equipment at existing local networks (Figure 1).

From the beginning it was intended that the server made to provide maximum flexibility, high reliability and a low price.

This led to the choice components used, mentioning specifically that server utilization is independent of the presence of the networks it is connected to a particular operating system, be it Windows, Linux, Mac OS X, Android or another; the server was thought, that it can work with any of these operating systems, which is possible by the design Atmel microcontroller ATMEGA 2560 and UC platform for manufacturers, but also network controller W5100.

2.2. Working principle

The client , connects to the Internet platform: PC, smart device, the Web server in question and hereinafter ATMEGA 2560 Web server, the algorithm is shown in Figure 2.

Based on a request it receives real-time information requested but at the same time can transmit commands terminated by validation or invalidation of input and output ports of the server, thus exerting control functions.

The Web server will connect it to the data network via the ethernet network block that provides a set of SPI command ports, a signalling unit with LED, to display Ethernet communication activity level and physical bidirectional port with RJ45 jack for connecting itself to support copper Internet network.

The network block is managed via command line interface SPI specific, meaning the microcontroller having server function will answer HTTP requests issued by network clients by generating HTML pages, pages that must contain information in real time "collected "from its ports, analog or digital. There are also regularly read logical status of ports or voltage values present at the input port pins corresponding analog or PWM output.

However, the server executes commands generated by the network client, commands that will be included in client request that can be like: state transition logic 1 or 0 digital output ports or the increment or the decrement the voltage of output PWM ports; maximum voltage applied to the microcontroller port is 5V.

3. Design Hardware Server's

3.1. Figures Block diagram of the server

The W5100 circuit [8] is an ethernet controller able to work at speeds of 10/100 Mb / s, designed for embedded applications characterized by ease of integration, stability, performance, low cost. W5100 has been designed to facilitate easy implementation connectivity of Internet, with TCP / IP protocol and Ethernet MAC & PHY without requiring an operating system.

The integrated module,TCP / IP , of the W5100 controller allows a wide range of Ethernet protocols: TCP, UDP), IP4V, ICMP, ARP, IGMP, and PPPOE, which have proved successful in various applications for many years.

The Ethernet controller block diagram [8] is shown in Figure 3.

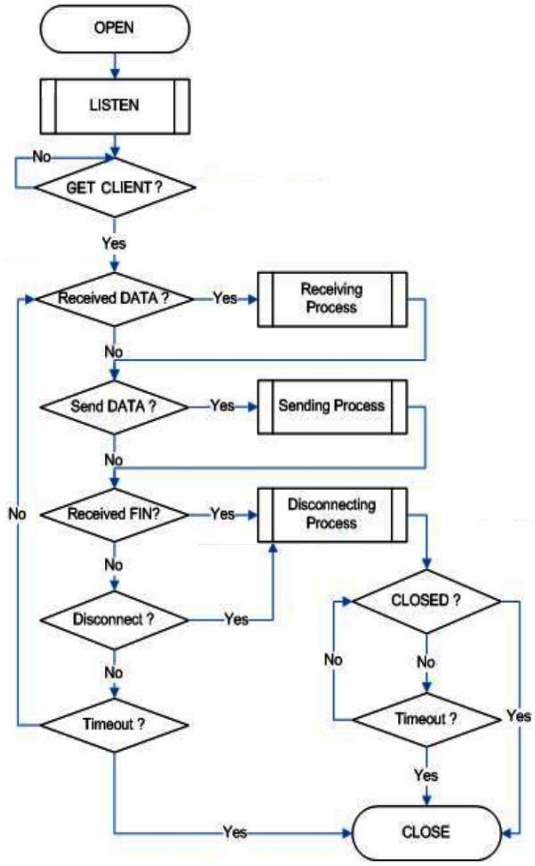

Figure 2. The communication algorithm, Web Server ATMEGA 2560 - user Server

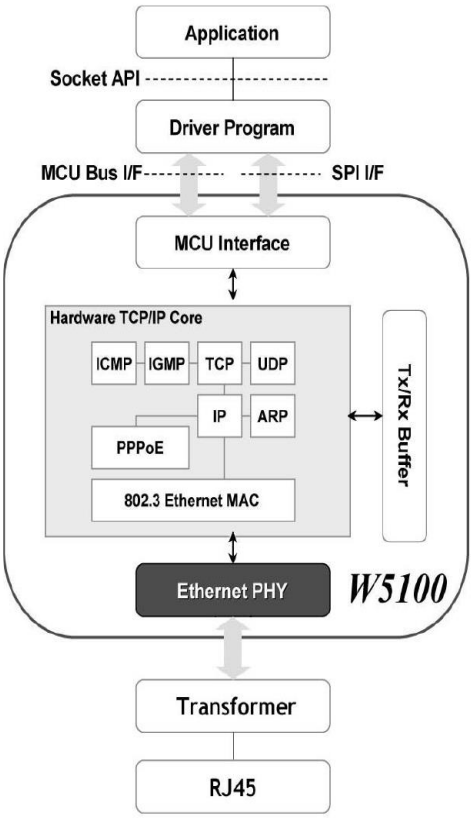

Figure 3. Ethernet Controller Block Diagram

3.2. Setting network parameters

Programming the network parameters [8], are set in the Main through four registers existing in the W5100 network controller. That registers intended network parameters each containing four bytes and the maximum value allowed is 255 byte and such resolution is 32-bit addresses for IPv4 domain.

The GWR register (Gateway IP Address Register), (Figure 4) sets up the default gateway address.

Example in case of "192.168.0.1"

The SUBR register (Subnet Mask Register), (Figure 5). sets up the subnet mask address.

Example in case of "255.255.255.0"

The SHAR register (Source Hardware Address Register), (Figure 6) designed for configuring the Ethernet physical interface MAC type and the default is "00.08.DC.01.02.03".

The SIPR register (Source IP Address Register), (Figure 7) set the most important value, to the ethernet interface, ie IP address, which initially contains a Class C address, public class, with the value: "192.168.0.3".

0x0001	0x0002	0x0003	0x0004
192 (0xC0)	168 (0xA8)	0 (0x00)	1 (0x01)

Figure 4. GWR register

0x0005	0x0006	0x0007	0x0008
255 (0xFF)	255 (0xFF)	255 (0xFF)	0 (0x00)

Figure 5. SUBR register

0x00	0x08	0xDC	0x01	0x02	0x03

Figure 6. SHAR register

0x000F	0x0010	0x0011	0x0012
192 (0xC0)	168 (0xA8)	0 (0x00)	3 (0x03)

Figure 7. SIPR register

For home users who have already created a local network, this class of IP addresses is ideal. A local network can be created by introducing a router to generate the LAN, beside the provider network of Internet, which is usually a network by type WAN.

Network addresses can be set to your choice by adding appropriate values for variables that keeps the field of network parameters.

3.3. Setting network parameters

The microcontroller used [6] (Figure 8) is a next-generation processor suitable for the intended purpose for this project in that:

- contains a UC fast enough, for the range of applications designed;
- has analog and digital ports, bidirectional, totaling 86 programmable line, which can be directly connected equipment that require ADC, PWM, digital communication series or parallel, in the range 0-5 V;
- allows a flexible dialogue, direct, indirect or SPI bus. This last option is used in project, to communicate with the W5100 circuit;
- internal memory is sufficient, 256 kB Flash,4 kB EEPROM, 8kB SRAM which can be supplemented with 64kB external;
- can be programmed in AVR Studio [9] or Arduino IDE 1.0.5 [10], languages "open source", followed widespread range of applications;
- with the W5100 circuit does not require a standardized operating system to communicate with "clients" who use Windows, Linux, Mac OS X, Android, etc.
- the program can be transferred via USB controller on a classic support;

4. Software Server's

The Web server software [11] is based on two programming environments, "HTML" and hybrid "C / C ++ and Arduino".

Environment "open source" Arduino makes it easy writing source code, the programmer by providing a package of standard libraries containing explained and illustrated instruction sets, memory for command and control, communication interfaces, LCD display drivers and so on.

It also provides programming environment compiling C language program and can achieve data transfer program in the microcontroller.

As an integrated development environment, the application Arduino IDE 1.0.5 version can run on platforms like Windows, Mac OS X, Linux and newer Android. He successfully supports ATMEGA 2560 microcontroller.

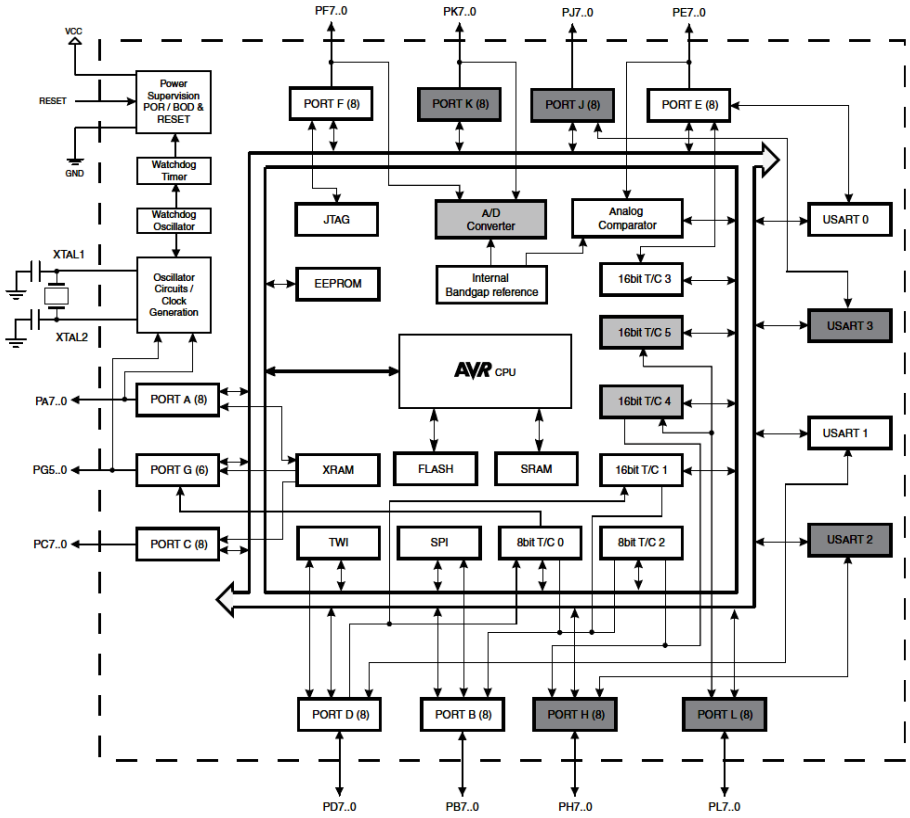

Figure 8. The block diagram of the ATMEGA 2560 microcontroller

4.1. Software interface

Scheme programming interface is built around a ATMEGA 16 microcontroller [12], ATMEGA 2560 microcontroller [6] positioned on the platform [5], which has the task of converting information from the USB port, the data communication interface compatible with the microcontroller (PE0 and PE1 ports) lines TxD RxD and according to the protocol or SPI programming ICSP port.

Program in source code is loaded into the programmer hardware software interface where the compiler is software as machine language and after connecting programming interface. The operation of the microcontroller and load the program, microcontroller accumulates information transmitted from software interface.

4.2. The source code

The program is edited in Arduino programming environment and contains a total of 730 lines of program.

The program can access additional instructions for SPI protocol, Ethernet interface and EPROM memory included in bookstores SPI.h, Ethernet.h and EEPROM.h.

```
#include <SPI.h>          // SPI command library instruction
#include <Ethernet.h>  // Ethernet interface library instruction
#include <EEPROM.h>  // library instruction EPROM
```

As soon as the libraries necessary to define a variable of type Buffer for data communications network,

```
// buffer size used to capture HTTP requests
#define REQ_BUF_SZ   15    // maximum capacity of 15 characters
```

Network parameters are defined

```
//  NETWORK PARAMETERS
byte ip[] = {192,168,1,110};              // IP address
byte gateway[] = {192,168,1,1};           // Gateway address
byte subnet[] = {255,255,255,0};          // network mask
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};                  //MAC address
EthernetServer server = EthernetServer(80);   // Ethernet port
```

Variables and constants are declared then:

```
//  DEFINING VARIABLES
char HTTP_req[REQ_BUF_SZ] = {0};          // reset
char req_index = 0;    // index in the buffer HTTP (req_index) starting from zero
char c = 0;
```

.....

Follow boot:

```
//  BOOT
void setup()
{
   Ethernet.begin(mac, ip, gateway, subnet);    //-boot ehternet interface
   server.begin();         // start "clients detect"

   for (int var=0; var < 8; var++){      //- reset Boolean area
    stareIesire[var] = false;
   }
   // loop declaration 8 output ports
   for (int var = 0; var < 8; var++){    //- setting selected as output ports
    pinMode(outDig[var],OUTPUT);      // output mode
    digitalWrite(outDig[var],LOW);       //-reset ports
   }
   // loop declaration of 8 input ports
   for (int var = 0; var < 8; var++){    //- setting selected as input ports
    pinMode(inDig[var],INPUT);         // input mode
   }
}
```

and loop program:

```
// LOOP PROGRAM
void loop()
{
  EthernetClient client = server.available();  // Check for cient!
if (client) {                                                        // get client?
      boolean currentLineIsBlank = true;
      while (client.connected()) {
       if (client.available()) { // data can be read from the client?
       char c = client.read();   // read one byte (character) of the client
    .....
```

In this section are essential *// Interpretation subroutines on client requests to access HTML pages as they are received from the client* interprets commands: *"spaţiu", "main"* , *"log"* , *"ret"* , *"por"* , *"sec"* , *// Interpretation and requests from the client as the port command is received from the client interprets commands type "outHk", "outLk" , k= 0...7*

Difficulty of the program is to process the information received from the client combined with information "collected" from the microcontroller peripherals.

Also, gaining control over the Internet server requires combining the two programming languages: Arduino (based on C / C ++) and HTML.

4.3. Communication between server and user

The communication between the server and the user is based on the establishment of codes by which to transmit text and execute commands (Table 1) for both control and ports for accessing pages provided by the server.

These codes will be assigned to the buttons on the main page and in the interaction with the user, issued codes will be attached in HTTP requests required the client to the server.

Table 1. Codes for the function of ATMEGA 2560 WEB Server

COD	FUNCTION	COD	FUNCTION	COD	FUNCTION
outH0	P.38 = HIGH	pwmH0	Inc. PWM.2	main	Main Page
outL0	P.38 = LOW	pwmL0	Dec. PWM.2		
outH1	P.39 = HIGH	pwmH1	Inc. PWM.3	log	Login Page
outL1	P.39 = LOW	pwmL1	Dec. PWM.3		
outH2	P.40 = HIGH	pwmH2	Inc. PWM.4	ret	Network Page
outL2	P.40 = LOW	pwmL2	Dec. PWM.4		
outH3	P.41 = HIGH	pwmH3	Inc. PWM.5	por	Ports Page
outL3	P.41 = LOW	pwmL3	Dec. PWM.5		
outH4	P.42 = HIGH	pwmH4	Inc. PWM.6	sec	Security Page
outL4	P.42 = LOW	pwmL4	Dec. PWM.6		
outH5	P.43 = HIGH	pwmH5	Inc. PWM.7		
outL5	P.43 = LOW	pwmL5	Dec. PWM.7		
outH6	P.44 = HIGH	pwmH6	Inc. PWM.8		
outL6	P.44 = LOW	pwmL6	Dec. PWM.8		
outH7	P.45 = HIGH	pwmH7	Inc. PWM.9		
outL7	P.45 = LOW	pwmL7	Dec. PWM.9		

The server will receive the characters in demand, will decode the codes and execute actions assigned to each code. To avoid any reading error codes in the program were not used repetition loops.

5. Implementation. Experiments

5.1. Logging into the server

The server identify potential client and server responds to requests initiated by the firm.

The first request initiated by the client will be connecting to the server. In response generates a web server, the client receives page to view and whose graphical form, corresponding to a standard login page (Figure 9) provided with boxes for placing the keyboard data Login: Username and password, plus a button to validate the data.

Managing data entered from the keyboard, not driven directly from the microcontroller server, which makes the web page content will be inserted login scrip JavaScript (Figure 10) with the task of verifying the data account, entered with the saved microcontroller.

The script shown in Figure 1, the program starts at line no. 4 and ends at line 24. In addition to the role of "guardian", this script is designed to generate alarms JavaScript in the login page by signalling with text messages (displayed in independent windows superimposed login page) login errors, errors that can be generated: account validation request without keyboard input data, account validation application without entering the password, or account validation request by entering false data (Figure 11).

Figure 9. Home Login

http://192.168.1.110/log - Original Source

File Edit Format

```
1  <!DOCTYPE HTML>
2  <html>
3  <head>
4  <script type="text/javascript">
5  function verifica() {
6  var username=document.getElementById("username");
7  var pass=document.getElementById("pass");
8  var isValid=true;
9  var utilizator = "Luci";
10 var parola = "1234";
11 if (username.value=="") {
12 alert("Introduceti un nume de utilizator!");
13 isValid=false;}
14 else if(pass.value=="") {
15 alert("Va rugam sa introduceti parola!");
16 isValid=false;}
17 else if(username.value!=(utilizator)) {
18 alert("Numele de utilizator sau parola sunt incorecte!");
19 isValid=false;}
20 else if(pass.value!=(parola)) {
21 alert("Numele de utilizator sau parola sunt incorecte!");
22 isValid=false;}
23 return isValid;}
24 </script>
```

Figure 10. The Login page script

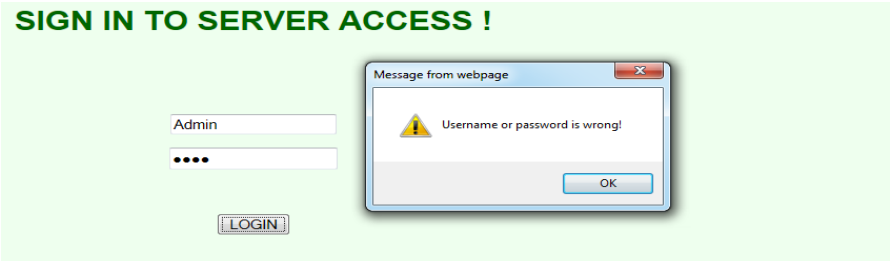

Figure 11. JavaScript Alert, false data logging

5.2. Main page

Home or "main page", provides the user with an interactive panel with a discount rate which is currently set at 10s. The value of the discount rate can be changed, of course.

It appears that the current application was designed for command, control and monitoring of 32 independent programmable lines, belonging to four 8-bit ports. Two ports are used for inputs or digital outputs; status of each line, 0 or 1 is indicated on the main page via a virtual LED assigned to each line. For output the settings for each line are buttons ON (logical 1), OFF (logical 0). Each of the 8 port analog input lines can be brought voltages in the range 0-5 volts through a successive approximation ADC with internal 10-bit ATMEGA 2560's are converted into digital signals by multiplexing. The eight analog output lines of the port are used to generate PWM signals ordering the idea of a three-phase static converter. Each output can be generated pulse train of amplitude U = 5V with variable duty cycle can be changed from the "+", "-".

The left panel shows a black band (Figure 12), which gives the user full rights auxiliary opening pages dedicated to amend additional names of ports, network configuration, modify data logging and the bottom of the strip is present output path or logout action.

Front panel made physically ATMEGA 2560 Server is shown in Figure 13.

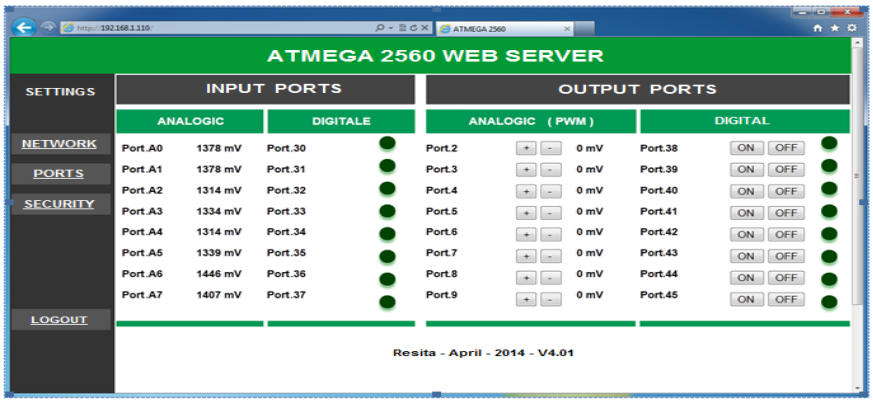

Figure 12. Main page of ATMEGA 2560 Web server

Figure 13. Front panel, ATMEGA2560 Server without LEDs

On the left side you can see the LEDs for communication control. In the central area is observed RJ 45 connector for interconnecting server - client via Internet and left 4 such connectors for interconnecting LAN server with the equipments/sensors.

Conclusions

The Web server with microcontroller ATMEGA 2560 has been designed for distance communication between manufacturing equipment's and/or sensors and users. It is part of the category of modern command, control and monitoring at distance entities and of different equipment and/ or sensors. The communication between equipment/ sensors on the one hand, and the user, on the other hand, is done by means of already classic networks computer: the server Web-user communication is connected via Internet, the server equipment/sensors communication is connected by LAN type networks.

The techniques used for realizing the server have covered a large spectrum of areas such as data transmission by networks communications, implementing interactive Web pages, programming and using items designed with programmable circuits, and generally also microcontrollers, especially in editing their programs, electric and mechanic compatibility of the server elements with the other physical components, using virtual systems.

The server permits communicating with other network components that they are part of, regardless of the operating system: Windows, Android, Mac X, etc. The source code can be written in open source Arduino or Studio.

For carrying out the proposed objectives, the authors have physically created the server using two platforms, an UC board and a network board and they have written the source code in open source language Arduino IDE 1.0.5., which at the moment contains 730 program lines.

The carried out tests, have shown that the program created for the server is correct, efficient and precise assuring the data acquisition, their processing and supplying calculation results on the graphic interface represented by web pages generated in HTML language.

Upon experimenting one has noticed the using the Firefox browser is optimal, lacking the unwanted effect of loading a page with a horizontal sweeping effect.

The carried out experiments have shown that the server is viable and has a stable functioning. The server is part of the "Low Cost" category, its hard components being very cheap, and the software dedicated to the used components is "free".

The signals generated and acquired by the server are signals working at the level of own ports with reduced values of tension and power, this is why it is useful to extend the signal levels, so as to be compatible with equipment's/sensors, fact finalized in the future.

One intends a development of the server [13]. Another direction is the security of our systems of the information transmitted through the network, in parallel with the optimizing of its functioning.

References

[1] Stinean A I, Preitl S, Precup R E, Dragos C A, Petriu E M and Radac M B 2013 *DOF control solutions for an electricdrive system under continuously variable conditions*, 8th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), Timisoara, Romania, May 23-25, pp 115-120

[2] Morosabn A D and Sisak F 2013 *An intelligent system designed for controlling the manufacturing process in a flexible manufacturing system*, 8th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), Timisoara, Romania, May 23-25, pp 353-357

[3] Vujovic V, Maksimovic M, Perisic B and Milosevic V 2014 *A Grafical Editor for RESTful Sensor Web Networks Modeling*, 9th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), Timisoara, Romania, May 15-17, pp 61-66

[4] Chioncel C P, Gillich N, Chioncel P and Gillich G R 2007 *CMS solutions in monitoring an real-time data transfer of photovoltaic plants,* XV-th International Symposium on Electrical Apparatus and Technologies SIELA 2007, Plovdiv, Bulgaria, May 31-June 1, pp 14– 17

[5] ***http://arduino.cc/en/Main/arduinoBoardMega2560

[6] ***http://www.atmel.com/images/doc2549.pdf

[7] ***http://www.dx.com/p/ethernet-w5100-shield-network-expansion-board-w-micro-sd-card-slot-for-arduino-152203#.VNLd7i4jWAQ

[8] ***http://www.wiznet.co.kr/Sub_Modules/en/product/Product_Detail.asp?cate1=5&cate2=7&cate3=26&pid=1011

[9] ***http://www.atmel.com/microsite/atmel_studio6/

[10] ***https://code.google.com/p/arduino/downloads/detail?name=arduino-1.0.5-windows.exe&can=2 , 2014B

[11] Berdie A D, Osaci M, Prostean G and Cristea A D 2011 *Web Programming features on integrated system SAP*, 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), Timisoara, Romania, May 19-21, pp 227-230

[12] ***http://www.atmel.com/images/ doc2466.pdf

[13] Sebu M L and Ciocarlie H 2014 *Applied process mining in software developement, Case study*, 9th IEEE International Symposium on Applied Computational Intelligence and Informatics, (SACI), Timisoara, Romania, May 15-17, pp 55-60