The implementation of a set of controls in an Arduino ATMEGA 2560 microcontroller system

CORINA DANIELA CUNTAN*, IOAN BACIU*, CAIUS PANOIU*, CEZARA-LILIANA RAT*

* Electrical Engineering and Industrial Informatics Department, Timisoara Polytechnical University, Faculty of Engineering Hunedoara, Revolutiei Street, no 5, Hunedoara, cod 331115, Romania,{ baciu.ioan, corina_c, c.panoiu}@fih.upt.ro)

Abstract: - This paper presents an application implemented on an Arduino development system using an ATMEGA 2560 microcontroller. The program controls ten outputs in sequential order and it is loaded onto the memory of the microcontroller. Each output is activated only when the system receives information about the previous command. These commands are highlighted by an optical LED display circuit which is provided with galvanic separation by optocouplers.

Key-Words: - microcontroller, optocoupler, sequential command, ARDUINO board

1 Introduction

A microcontroller is a circuit made on a single chip, typically containing: the central processing unit, the clock generator (to which a quartz crystal must be added from outside) or, in less demanding applications, a RC circuit), volatile memory (RAM), nonvolatile memory (ROM / PROM / EPROM / EEPROM), I/O serial and parallel devices, interrupt controller, DMA controller, counters/timers, A/D and D/A converters, etc., peripherals. With a MC, we can realize an integrated controller (Embedded Controller, EC). [1]

An embedded controller is part of a system built for a specific purpose, other than usual general calculations. Besides the MC, an embedded controller requires additional hardware to perform its function.[2]

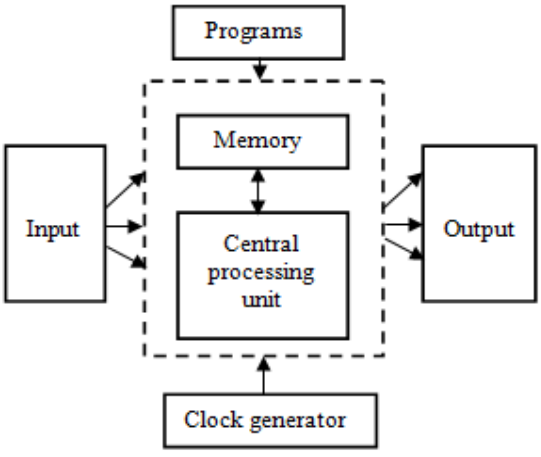

Fig. 1 Simplified diagram of a microcontroller

A microcontroller can be defined based on its simplified representation. (Fig. 1)[2],[3]

As inputs, we usually use signals from individual switches or from transducers (temperature, pressure, photo, specialized transducers). The inputs can be digital or analog.

The digital inputs convey discrete signals, the information "read" being the information to be sampled when reading that line.[4] The analog inputs convey the information expressed by continuous functions of time. The "reading" of them by the microcontroller requires the presence of circuits able to process this information, i.e. analog comparators or analog-to-digital converters, whose outputs are read by the MC.[5]

The outputs can be analog, in which case they actually represent outputs of the analog-to-digital converters, or digital, in which case the information is generally stored on them until a new entry is operated by the UC at a port of the MC. The outputs can control display devices, relays, motors, loudspeakers, etc.[2],[6]

2 Theoretical Issue

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.[7]

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or a battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack.

ISBN: 978-960-474-328-5

Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. [7]

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. [7],[8]

The power pins are as follows:

- VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). Voltage can be supplied through this pin, or, if it is supplied via the power jack, it can be accessed through this pin.

- 5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board. [7]

- 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

- GND. Ground pins.

- IOREF. This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs for working with the 5V or 3.3V. [7]

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip. [7]

- External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. [7]

- PWM: 2 to 13 and 44 to 46. Provide 8-bit PWM output with the analogWrite() function.

- SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.

- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

- TWI: 20 (SDA) and 21 (SCL). Support TWI communication using the Wire library. These pins are not in the same location as the TWI pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values).

By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- AREF. Reference voltage for the analog inputs. Used with analogReference().

- Reset. If this line is LOW it resets the microcontroller. Typically used to add a reset button to shields which block the one on the board. [7]

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication.

An ATmega16U2 (ATmega 8U2 on the revision 1 and revision 2 boards) on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically.

The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2/ATmega16U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1). [7], [8]

The Arduino Mega can be programmed with the Arduino software.

The ATmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. [7], [8]

The paper presents a program developed in C burned onto the memory of the microcontroller that writes the control sequence to the output pins only after first receiving a feedback showing that the previous command was executed. (fig.2) [7], [8]

ISBN: 978-960-474-328-5

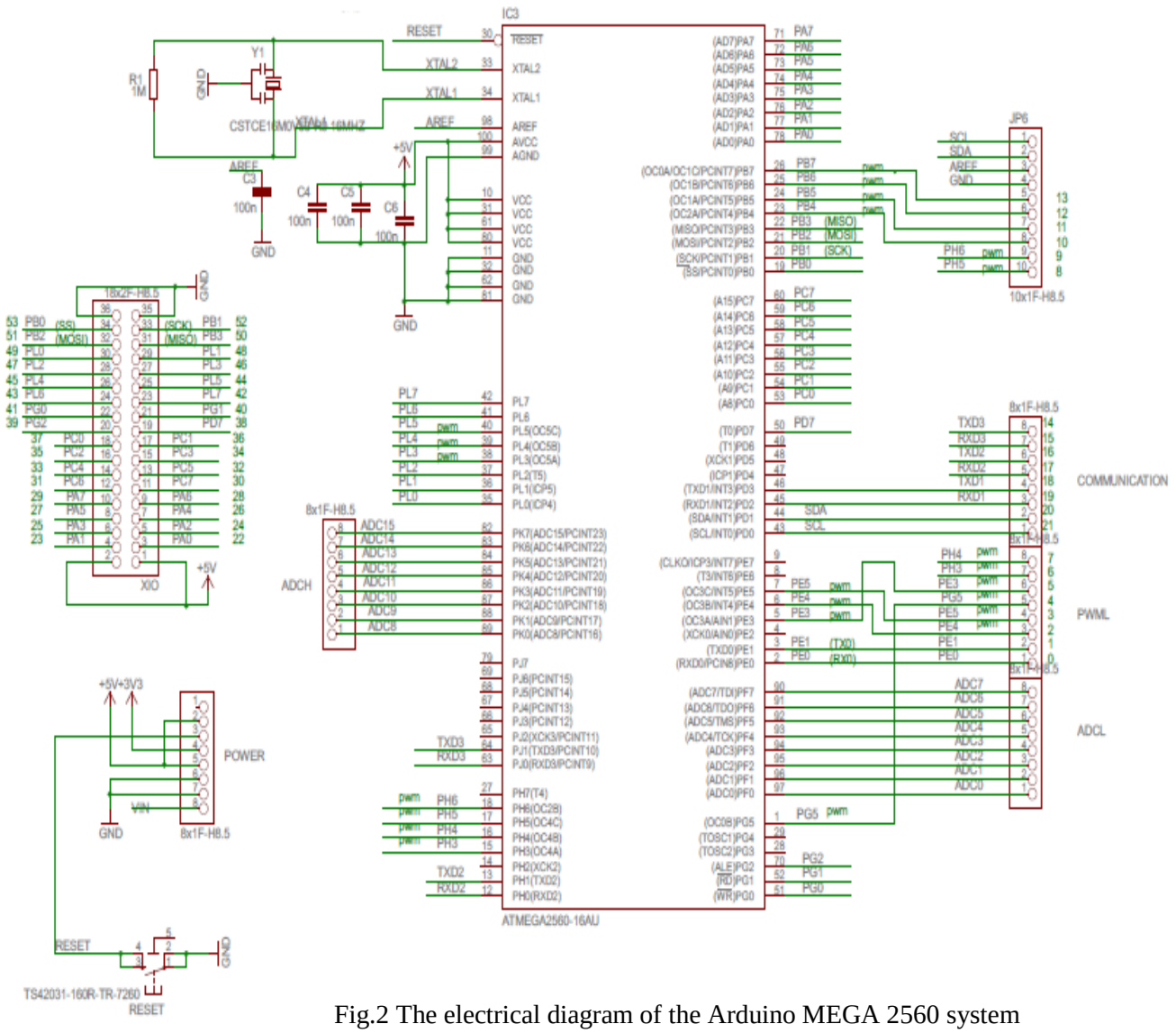

Fig.2 The electrical diagram of the Arduino MEGA 2560 system

The program uses the digital pins from 0-9 as output pins, and those from 30-39 as input pins. It doesn't matter which of the pins are used, as any of the digital pins on the board can be set either as input or as output.

It is necessary to associate each input pin to an output pin in order to verify the execution of the previous command, so as to make sure that each command was sent successfully. The delay between the lighting of a LED by writing an output pin and reading the LED status by reading the corresponding input pin, is 1s = 1000ms.

Both the input as well as the output pins are retained as vectors. They are initialized in the array in ascending order, but the order can be random, taking only into account the correlation between input pins and output pins.

At first, all pins are initialized to HIGH. The command to the LED coming through an inverter, the effect is that of turning off all of the LEDs.

Then, in a WHILE loop, the pins are reset one at a time, the LEDs lighting up gradually. After writing each pin, the system waits for an amount of time equal to the timer and reads the corresponding input pin state, so to check if the LED was really on. If it reads a logic 1, the LED was lit and the system can continue with turning off the current LED and lighting up the next LED. If it reads a logic 0, the loop is reversed beginning with the first LED.

Power supply to the microcontroller is done through the computer connection cable.

The commands highlighted by the LEDs can be transmitted to execution elements by means of suitable circuits.

The control sequences obtained from the output pins of the board are brought to the input of the galvanic separation circuit through the inverter circuit in order to increase the control current. [fig.3][9]

ISBN: 978-960-474-328-5

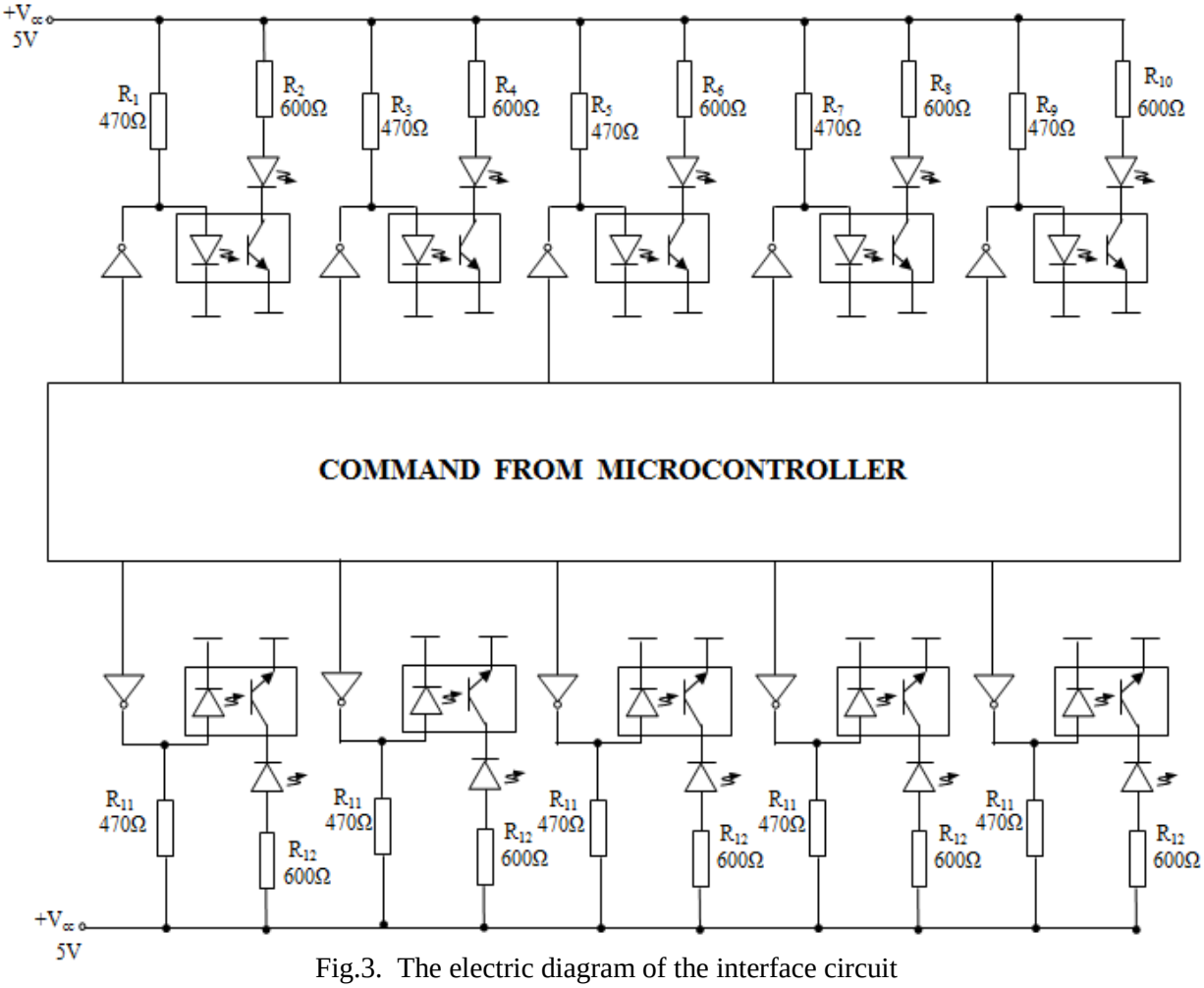

Fig.3. The electric diagram of the interface circuit

Below is presented the program that is burned on the memory of the microcontroller.

```
int timer = 1000;           // The higher the number, the slower the timing.
int ledPins[] = {
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };       // an array of pin numbers to which LEDs are attached
int pinCount = 10;           // the number of pins (i.e. the length of the array)
int inPins[] = {
30, 31, 32, 33, 34, 35, 36, 37, 38, 39 };
int ok = 1;
void setup() {
// the array elements are numbered from 0 to (pinCount - 1).
// use a for loop to initialize each pin as an output:
for (int thisPin = 0; thisPin < pinCount; thisPin++)
{
pinMode(ledPins[thisPin], OUTPUT);
    }
for (int thisPin = 0; thisPin < pinCount; thisPin++)
{
pinMode(inPins[thisPin], INPUT);
    }
  }
void loop() {
for (int thisPin = 0; thisPin < pinCount; thisPin++) {
digitalWrite(ledPins[thisPin], HIGH);
    }
ok = 1;
int thisPin = 0;
while (ok == 1)
{
digitalWrite(ledPins[thisPin], LOW);
delay(timer);
digitalRead(inPins[thisPin]);
if (inPins[thisPin])
ok = 1;
else
  ok = 0;
  digitalWrite(ledPins[thisPin], HIGH);
  thisPin++;
  if (thisPin >= pinCount)
  ok = 0;
  delay(timer);
  // turn the pin off:}}
```

By changing the parameters of the program it is possible to change the frequency of the control signals. If desired, the number of controled items can be increase by changing the number of pins in the program to the maximum number of output pins of the ARDUINO development system.

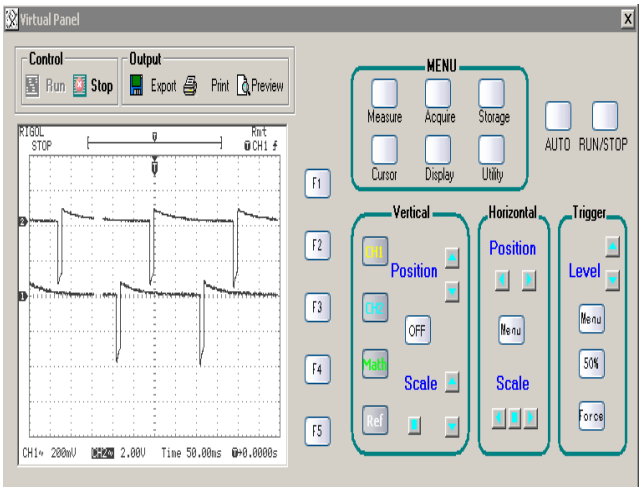

Fig.4 Waveforms at a frequency of 5 Hz

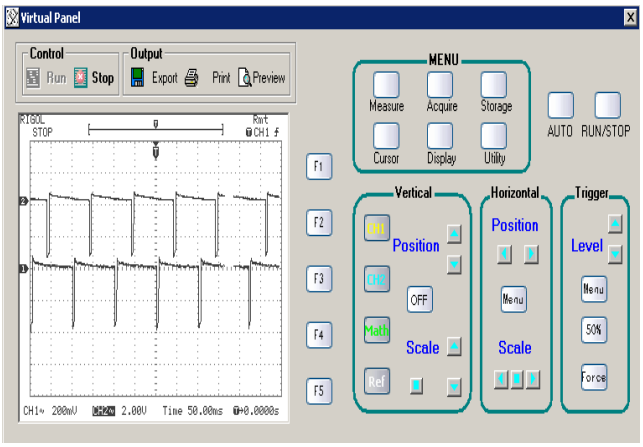

Fig.5 Waveforms at a frequency of 10 Hz

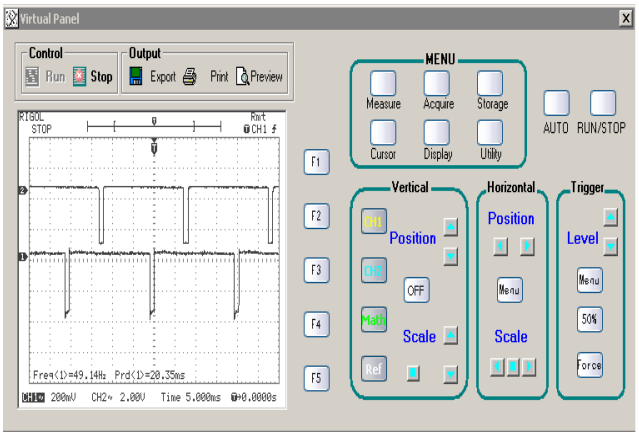

Fig.6 Waveforms at a frequency of 50 Hz

The highlighting of the command sequences is obtained through a digital oscilloscope that allows the transfer of data to the memory of a computer. The following waveforms are obtained on two output pins at a frequency of 5 Hz (fig. 4), 10 Hz (fig. 5) or 50 Hz (fig.6).

Experiments were made at low frequencies so that a human operator can directly view the functioning of the system.

The experimental setup (fig. 7) shows just such a situation.

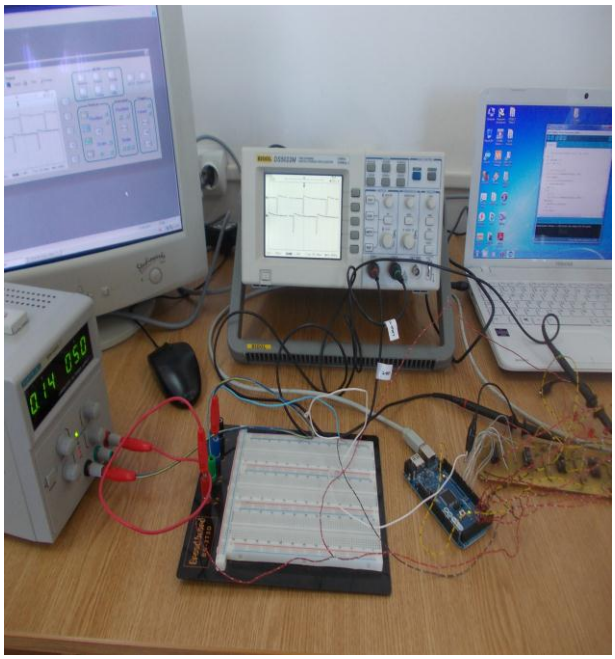

Fig.7. The experimental model for visualizing the sequential command

3 Conclusion

By using the Arduino board with the ATMEGA 2560 microcontroller, programming is made relatively easy.

This application controls a series of switching elements allowing the inter-conditioning of the controls. In this case we chose a sequence of 10 commands that can be visualised optically through the LEDs and that are sent one after another but only after receiving information through which the execution of the previous command can be deduced.

If one of the commands hasn't been executed, the command sequence starts from the beginning.

The control frequency can be modified within the limits of the Arduino board and also the number of commands can grow to the maximum number of output pins.

ISBN: 978-960-474-328-5

References:

[1] Pănoiu Manuela, Muscalagiu Ionel, Pănoiu Caius, Raich Maria, *Educational Software for study the Performances of some known Parallel Algorithms*, Wseas Transactions on Information Science and Applications, nr 10 v7, 2010, pp. 1271-1283;

[2] http://www.unitbv.ro/faculties/biblio/interfete_specializate/curs.pdf;

[3] Pănoiu Manuela, Iordan Anca, Pănoiu Caius, Ghiorghioni Loredana, *Interactive Environment For Learning The Rules For Physical Storing Of The Information In The Computers Memory*, Proceedings Of The 8th WSEAS International Conference On Education And Educational Technology (Edu09), Genova, Italy, October 17-19, 2009, 239-244;

[4] Iordan Anca-Elena, Pănoiu Manuela, Muscalagiu Ionel, Rob Raluca, "*Realization of an Interactive Informatical System for the Quadric Surfaces Study*", Proceedings of WSEAS International Conference on Computers, 2009, Rodos, Grecia, pag. 205-210, ISBN: 978-960-474-099-4;

[5] Abrudean Cristian, Pănoiu Manuela, *Software Application Implement in Java for Electrical Lines Dimensioning*, Proceedings of the 8th Wseas International Conference on Applied Computer Science (acs'08) , 21-23 Noiembrie, Venice, Italy, pp. 614-619;

[6] Iordan Anca-Elena, "Development of Interactive Software for Teaching Three-Dimensional Analytic Geometry", *Proceedings* of WSEAS International Conference on Distance Learning and Web Engineering, 2009, Budapesta, Ungaria, pag. 23-28, ISBN: 978-960-474-115-1;

[7] http://arduino.cc/en/Main/arduinoBoardMega2560;

[8] http://www.arduino.cc/en/Tutorial/Array;

[9] Popa Gabriel Nicolae, Diniş Corina Maria - Digital Measurement Of The Voltage And The Temperature In The Sections Of Industrial Plate-Type Electrostatic Precipitators, Advances In Systems Theory, *Signal Processing And Computational Science*, *Proceedings Of The 12th WSEAS International Conference On Signal Processing, Computational Geometry And Artificial Vision (ISCGAV '12)*, Istanbul, Turkey, August 2012, 21-23, ISBN: 978-1-61804-115-9, pp. 94-99;