Using microcontrollers for orienting a set of mirrors to focus the light beam

C Silaşi¹, C Cunţan² and I Baciu³

¹ "Key Safety Romania", Ribita Plant
² Politehnica University of Timisoara, Department of Electrical Engineering and Industrial Informatics, 5 Revolution Street, 331128 Hunedoara, Romania

E-mail: corina.cuntan@upt.ro

Abstract. This paper presents an independent control system for two mirrors tracking the position of the sun and allowing the focalisation of sunlight to increase the efficiency of the energy conversion process. The application is designed to operate on the principle of a master that controls slave devices. The mirrors are designed to be independently manoeuvrable, this being possible by using microcontrollers to handle the management of their local orientation. For orienting a surface in order to capture or reflect the solar radiation, it is necessary to know the sunlight direction with a very high accuracy.

1. Introduction

The microcontrollers are VLSI type integrated circuits (VLSI = Very-large-scale integration), incorporating most of the components required to build a computer microsystem: central processing unit (CPU), program memory, data memory, interrupting system, digital input/output ports, analogue to digital converters, digital to analogue converters, counters and timers, serial network interfaces, and others. In many cases, a single circuit solves all the monitoring and control functions of an application [1].

Besides the microcontroller, an application requires a program to execute, and other few elements that constitute a logical interface with the stabilizing elements [2]. The programming can be done in several languages, such as: Assembler, C and Basic, which are the mostly used programming languages. The Assembler belongs to the low-level languages that are slowly programmed, but use the smallest memory space and give the best results in terms of program execution speed.

The programs in C language are easier to write, easier to understand, but are slower in execution than the programs in Assembler language. Basic is the easiest to learn and its instructions are the closest to human thinking, but, just like the C programming language, it is slower than Assembler.

After the program is entered into the microcontroller, this one must be connected to the application module. Based on its clock, the microcontroller executes the program instructions. Once connected, the microcontroller executes the recognition instructions from the beginning of the program, and then starts the actual operation phase. The circuit operation depends on many parameters, of which the most important are the ability of the hardware developer and the programmer experience in obtaining the best results when using the designed program [3].
A microcontroller-based system allows obtaining a maximum power at a given time, by using a solar orientation system of certain systems of reflection and focalization of the solar radiation.

The solar energy is a green energy source, free and inexhaustible. With the help of solar radiation collection systems, the solar energy can be converted into electricity or heat, which can be used for own use, or injected into the public network. The technological development and, at the same time, the growing price of energy in general, justifies the interest of an increasingly number of users to purchase energy production / capture systems.

2. Presentation of the paper
This paper deals with the design and implementation of a solar radiation focalisation system, as basis for a range of household and industrial applications.

The use of microcontrollers in industrial processes has a high dynamics in the current period, because of their advantages over the other data processing & capture solutions.

For these reasons, we describe in this paper the use of microcontrollers in the management and control of a focalisation system used for the solar radiation incident on a central point, for further applications of the focused energy.

This can be done by using measuring devices able to direct the sunlight, or algorithms that calculate the direction of solar radiation based on various models developed over time.

For finding the direction of solar radiation, we chose the method described in “Solar Position Algorithm for Solar Radiation Applications”, because it allows the calculation with uncertainties of ±0.0003°, in the period from the year -2000 to 6000 [4].

The determination of the vector that defines the direction of solar radiation, hereinafter referred to as the sun vector, allows us to calculate the orientation of the reflective surface as being perpendicular on the bisector of the angle formed by the sun vector and the vector connecting the centre of the reflective surface and the centre of the outbreak.

In this application, the PC program of interfacing with the central microcontroller sends the following data to the microcontroller: year, month, day, time, minute, second, time zone (negative in relation to Greenwich meridian), longitude, latitude, height at which the system is located, atmospheric pressure and ambient temperature. Based on these data, the microcontroller calculates the solar radiation vector and sends it to all the network microcontrollers, along with the outbreak coordinates. The slave microcontrollers calculate the mirror-outbreak direction and place the mirror perpendicular to the bisector of the angle between the two vectors.

**Figure 1.** Block diagram of the orientation system of an array of mirrors.
The application has three main components (Figure.1):

**The PC application for system monitoring and control** is carried out in C#, and is intended to provide the system’s interface with the user. The application runs on a normal PC and communicates with the central microcontroller by USB. The main role of the application is to keep the main system configuration data and to provide them to the central microcontroller at the user’s command or at regular intervals, according to the settings.

**The application running on the central system** (central microcontroller) has the following functions:
- to manage the communication with each client and the PC application;
- to calculate the sun vector to be sent to the clients;
- to store and send the outbreak position to the clients.

**The application running on the client system** (slave microcontroller) has the role to calculate the mirror orientation based on the data received from the central microcontroller and to effectively manage the mirror rotation with two stepper motors.

Regarding the practical implementation solution, we chose as master the ATXMEGA256 A3 microcontroller[5]. (Figure 2)

This microcontroller has a number of features that make it suitable for this application. It is an advanced microcontroller, has enough memory (256k) to run such an application, has sufficient communication resources to be able to manage the communication with a large number of clients and, in the same time, it has peripherals that allow the acquisition of local information, as follows: temperature, atmospheric pressure, GPS coordinates from specialized devices or sensors, if required.

Also, we chose to use a development board, built around the AtMEGA256, of ATxmega AL-XSLED_EXT type, due to the fact that this board implements a standard solution for using the microcontroller, in terms of energy supply, and has also a solution for the USB communication, i.e. a mini USB connector for communication with the application installed on the PC, and two programming interfaces, JTAG and ISP.

![Figure 2. The application operation diagram.](image-url)
Another advantage of using a development board is its solid construction, electrically and mechanically tested. All these features make the chosen solution to be suitable for the intended use.

The master board has the port USARTC1 configured as communication port with the clients. The port USARTD1 is connected to the USB interface by default, reason why it is used as a connection with the PC application.

The port USARTC1 uses the pins 6 & 7 and RX & TX, respectively. The pins TX and RX of the clients will be connected to those pins.

The USART communication is conceived as point to point bidirectional communication between two participants in communication.

This method requires the creation and implementation of a protocol to provide each client with its own identification number, which is going to be used as a key for communication with that client.

Also, the master keys can be configured so that, if such a key is received, all the clients react to the received information; these are the so-called broadcast messages. In this configuration, the communication will be made only from the central microcontroller to the clients.

The chosen slave microcontroller is the model AtMega328. This microcontroller has a storage space of 32k, sufficient to host the application to be run, it has input-output ports that can be used to control the two stepper motors required for the reflector orientation, and serial communication resources for integration into the network containing the central microcontroller [3], [6].

In a similar way as the master microcontroller, we can use a development board built around the AtMega328 microcontroller, of MINI-AT type, board that implements a standard solution for using the microcontroller in the circuit. The board has a USB interface for programming and communication with a PC, and a number of connectors placed on the sides of the board to which all the pins of the controller are connected, so that the device can be easily used on prototyping boards.

We use stepper motors having the following features: number of steps per revolution: 200, winding current: 2.6A, and supply voltage: 5V. The motors develop sufficient torque to be able to rotate the reflector safely (Figure 2).

The actual stepper motor control is carried out with motor drivers LMD18245 that can withstand a current of 3A and a voltage of 55V. This type of driver has inputs that require an input voltage level of 5V, which allows us to use it with a microcontroller that has an output voltage level of 5V.

As regards to the driver, 5 output ports are required for each motor. Therefore, we need 6 output ports to control two motors.

The driver inputs are controlled by the outputs of the microcontroller, as follows:
- The port D2 controls the current direction in the winding A of the motor 1;
- The port D3 controls the current direction in the winding B of the motor 1;
- The port D4 controls the braking input, which stops the current passing through the motor windings 1;
- The port C0 controls the current direction in the winding A of the motor 2;
- The port C1 controls the current direction in the winding B of the motor 2;
- The port C2 controls the braking input, which stops the current passing through the motor windings 2.

In addition to these ports, the drivers need a reference voltage and a 4-bit codification for the voltage to set the current through the windings.

The reference voltage is 5V, and can be taken from the source. I chose to statically treat the determination of the current through the windings, based on experimental tests, by connecting the bits to the 5V supply voltage.

2.1. Programming

The programming has three distinct components:
- The program running on the PC and provides the user interface;
- The program running on the master controller;
- The program running on the slave controller.
The program running on the PC is made using C# in the programming environment Microsoft Visual Studio 2010, and consists of a graphic interface, whose main elements are: an area that includes the serial communication port settings, an area that includes the reflectors array settings, and an area of buttons for commands (Figure 3)[7].

As may be noticed, in the “SPA setting” area you can enter the data required for the calculation algorithm for the solar radiation direction vector.

![Figure 3. Graphical user interface for the program running on PC.](image)

These data are read at startup from a “config.cfg” file (Figure 4) placed in the same location as the executable program, given that most data required by the algorithm are specific to the geographical location and can be considered constants.

![Figure 4. Data required.](image)
When closing the program, the data are saved in the file. The exceptions to the above are the fields containing the current date and time, these ones being obtained at regular intervals from the system on which the application is running. These settings can be changed if unticking the checkbox on the right of each field. By ticking the check-box, the default value of the field is changed for the next loading of the program (Figure 3).

These data are sent only once to the central microcontroller, or every time when a parameter is changed, by using the Setup (Initialization) button.

In the second page of the application, we define the positions of all the clients relative to the same reference system as the outbreak. To initialise the network of reflectors, we must send these data to the central microcontroller, which, in its turn, sends to the clients the location of each one.

This information is sent only once, at the initialization of the system; then, it can be sent on request whenever any changes occur.

After initialization, the application starts to send, at regular intervals, the date and time to the microcontroller, these being the only data still required by the microcontroller.

The program of the central microcontroller is written in the C programming language. For its development, we used the AtmelStudio application, which is the integrated development environment provided by the microcontroller manufacturer Atmel.

Basically, the application running on the microcontroller calculates the solar radiation direction vector (based on the data received from the PC) and the current date and time, and sends this vector, along with the outbreak location, to the clients.

The microcontroller program is written in the C programming language; for its development, we have also used the AtmelStudio application, which has the role to receive data from the central controller, the solar radiation direction vector, the outbreak position and its own coordinates. Taking into account all these data and the current orientation of the mirror, it shall act accordingly.

The communication protocol involves the use of simple rules ensuring the transmission of information from the PC to the central microcontroller, and further to the client microcontrollers.

For easing this, the messages are organised as strings with a maximum length of 20 characters. A message is considered to be complete when it has 20 characters, or when the character “:.” is added to its end (if it has less than 20 characters) (Figure 5).

![Figure 5](image_url)

The commands have three characters in length for the central microcontroller and five characters for the clients, and are interpreted by a program that initially converts the three / five characters into an integer using a string hash function, to be able to use the SWITCH-CASE instruction for interpretation.

The mechanical part of the application (Figure 6) consists of a footed stand to which the azimuth axle motor is attached. The motor drives a gear wheel on which the zenith axle is mounted on two stands, axle which is driven by another motor. The reflector is fixed on this axle, by means of two stands.
The design is made in Solid Works, a CAD application for industrial design, and the parts are cut from 10 mm thick polyamide, using a CNC water jet cutting machine, the metal shaft is made of steel machined on a lathe, and the bearing housings are made of textolite sheet, machined on rotary cutter mill and lathe.

![Figure 6. Computer-aided design (CAD) of the mirror orientation system.](image)

3. Conclusions

For the practical implementation of the system, we need to create the PC programs, to build the central microcontroller and client microcontrollers, to assemble the mechanical parts and electronic circuits, and to check the application operation. (Figure 7)

For programming, we firstly realise the motor command, meaning the connection of the drivers to the control board, followed by the operational testing. The next step is to develop the communication part and to test its operation.

The string of characters received has been experimentally obtained after a series of trials, using a PC serial communication utility that sends commands to the microcontroller, followed by setting the microcontroller to send back to the PC the decomposed string, for checking it.

![Figure 7. Assembling and connecting of the electronic components.](image)
The sent string can be seen at the bottom of the window, and the decomposed response at the top of the window. (Figure 8)

To improve the communication between the clients and the central microcontroller, we can realise a bidirectional network to enabling the central microcontroller to take information from the clients and make the right decision.

![Figure 8. Testing the communication with the client.](image)

**References**

[1] ***http://www.unitbv.ro/faculties/biblio/interfete_specializate/curs.pdf
[2] Sebestyen Gh 2006 *Informatică Industrială* [Industrial Informatics], Publisher: Editura Albastră, Cluj Napoca, Romania
[3] Susnea I and Mitrescu M 2005 *Microcontrollers in Practice (Springer Series in Advanced Microelectronics)*, Springer, Berlin, Heidelberg, Germany
[4] ***Solar position algorithm for solar radiation applications by Ibrahim Reda and Afshin Andreas, available on the U.S. Department of Energy website http://www.nrel.gov, National Renewable Energy Laboratory of the Office of Energy Efficiency and Renewable Energy (EERE)
[5] Xavier F C 2000 *An introduction to the industrial applications of microcontrollers*, 3rd Conference On Physics And Industrial Development: Bridging The Gap (Copid 2000), September 4-7, Durban, South Africa
[6] ***http://retele.elth.ucv.ro/Bratu%20Cristian/MAP/002%20-%20Curs%20002%20-%20MAP%20-%20Microcontrollerul%20.pdf
[7] Zachariadou K, Yiasemides K and Trougkakos N 2012 A low-cost computer-controlled Arduino-based educational laboratory system for teaching the fundamentals of photovoltaic cells, *European Journal of Physics* **33**(6) 1599