No. 111/19, 17–23
ISSN 2657-6988 (online)
ISSN 2657-5841 (printed)
DOI: 10.26408/111.02

Submitted: 05.07.2018
Accepted: 27.08.2018
Published: 23.09.2019

DESIGN AND CONSTRUCTION OF A STARTUP SYSTEM FOR LEARNING TO PROGRAM MICROCONTROLLERS

Emilian Świtalski[1], Dorota Rabczuk[2]*

[1,2] Gdynia Maritime University, Morska 81-87, 81-225 Gdynia, Poland,
Faculty of Electrical Engineering, Department of Marine Telecommunications
[2] e-mail: d.rabczuk@we.umg.edu.pl, ORCID 0000-0003-0636-0464
* Corresponding author

Abstract: The article presents the design and execution of a startup system based on an Atmel AVR microcontroller. Due to its intended educational use, the system was equipped with a large number of peripherals connected to various microcontroller outputs using pin-cables. The diagram and PCB design feature an ergonomic layout of the microcontroller outputs and peripherals, to ensure clarity of connection. Libraries and examples were written in the C programming language, the code was compiled using Atmel Studio 7 software – the official Atmel programming environment. The startup system provides optimum conditions for learning related to microprocessor technology in the laboratory.

Keywords: AVR, Atmega, Atmel, microprocessor, devboard, eagle, pcb, embedded systems.

1. INTRODUCTION

The commercial range of startup systems for Atmel AVR microcontrollers is broad, although it is not fully suited for the concept of practical learning of microcontroller programming that we promote. According to this concept, an educational startup system should be equipped with a large number of external elements installed directly on the system's PCB, but not permanently connected to the microcontroller's lines. This provides the learner with an opportunity to perform simple installations, while relieving them of the difficulties of correctly tracing the power supply for the components, the need to suspend lines, etc. A review of the commercially available solutions shows that startup systems generally have a low number of external elements on the PCB [*Microboard for AVR 64-pin Manual*]. Systems with external elements permanently connected to selected microcontroller outputs without any modification possible are also available [*AVR Butterfly Evaluation Kit User Guide*]. The concepts closest to our own idea are those implemented in the ATB 1.05A [Kardaś 2016] and GrandEVBavr systems [*GrandEVBavr płyta ewaluacyjna dla mikrokontrolerów*

avr – instrukcja użytkownika]. GrandEVBavr provides the ability to select a microcontroller from the AVR ATMega family, although we decided that educational reasons do not justify the need to change the microcontrollers in the laboratory, and have limited the project to a single microcontroller (ATMega32).

The principle of the project is to use the Atmel Studio programming environment, available as freeware, and to program microcontrollers through direct registry entries. This programming method is versatile and can be performed in a similar manner for microcontrollers from different manufacturers. Based on the above assumption, we excluded the use of the Arduino platform libraries from educational use, as they are generally not incorporated in programming environments other than Arduino [Rabczuk 2011; Pałczyńska Rabczuk and Fornalski 2017].

The design and execution of a startup system was also the subject of an engineer's thesis [Świtalski 2017].

2. SELECTION OF THE EXTERNAL ELEMENTS OF THE STARTUP SYSTEM

The startup system was equipped with 16 LEDs, grouped in two rows of 8 diodes each, 8 microswitches, and a rotary encoder. This high number of simple elements stems from the fact that basic skills, such as register operations, use of different numerical systems, optimum use of logical, arithmetic and binary operators, as well as using conditional instruction and loops, is learned specifically using these elements. The freedom to perform connects enables switches to be connected on external interrupt lines and the functioning of interrupt procedures with the main program observed.

The startup system is equipped with external elements freely connectible to the microcontroller's lines by vertical cables, including: four 7-segment displays, a potentiometer with a connected slider, joystick, analogue keyboard, an H bridge made of bipolar transistors for connecting motors, an alphanumeric LCD, infrared receiver, triaxial accelerator-gyroscope, single and serially controlled sets of RGB diodes and a piezoelectric buzzer.

As startup system users can be beginners in learning microcontroller programming, a microcontroller with a DIP casing was used, equipped with a stand enabling damaged integrated circuits to be replaced. The ATMega32A microcontroller was chosen, with 40 external outputs available on vertical strips.

Power is supplied through a USB connection and through bolted connections (user's choice). All bolted connections are versatile, and can serve as outputs or inputs for any signal.

The USB adapter installed on the board provides a serial connection with a virtual PC port, and enables use of any terminal software for serial two-way

communication. The USBasp programmer connected through an SPI bus was selected to upload the executable program to the microcontroller's memory, as it integrates easily with the Atmel Studio environment and additionally can also supply power to the system, eliminating the need for an additional USB cable.

3. ELEMENT AND CONNECTION LAYOUT CONCEPT ON THE PCB

The element and connection layout on the board was planned by taking into account the ergonomic use of such a laboratory station. The elongated shape of the board enabled the inputs to be arranged along the bottom and left edges, with the external outputs on the top edge of the board. The internal connections between the microcontroller and its peripherals are located in the middle. With this solution, the connections do not cover the board's key elements, and manipulations performed on the adjustable elements do not obstruct the observations. During a lecture, a board image can be recorded and displayed in real time, or saved to a file without fear that important elements of the board may be obscured. The freedom in selecting connections on the board between the microcontroller and external peripheral systems ensures visual order for the installation.

The best solution for beginners is to independently make all electric connections, although some elements are difficult to install on the contact board, while others, when installed in this manner, are inconvenient to use. Given these limitations, a compromise is to execute all connections on the board, except those that will be connected to the microcontroller. In this way, the user has complete freedom in using the microcontroller's pins, which is important during learning.

The startup system is designed so that there are no configuration jumpers (other than the power supply jumper on the USBasp programmer), which, combined with the connection descriptions etched on the board, makes user documentation almost completely unnecessary. This approach imposes certain limitations on the choices available (microcontroller, power supply, etc.), but at the same time it is an encouraging simplification – one can focus directly on the planned project.

Care was taken to print descriptions on the PCB, so that anyone using the system faces no uncertainties concerning to what outputs the microcontroller's pins are connected, without the need to check the connections with a measuring device or refer back to the catalogue note. In the boards intended for do-it-yourself assembly, the element outlines and additional markings contribute to minimising assembly errors.

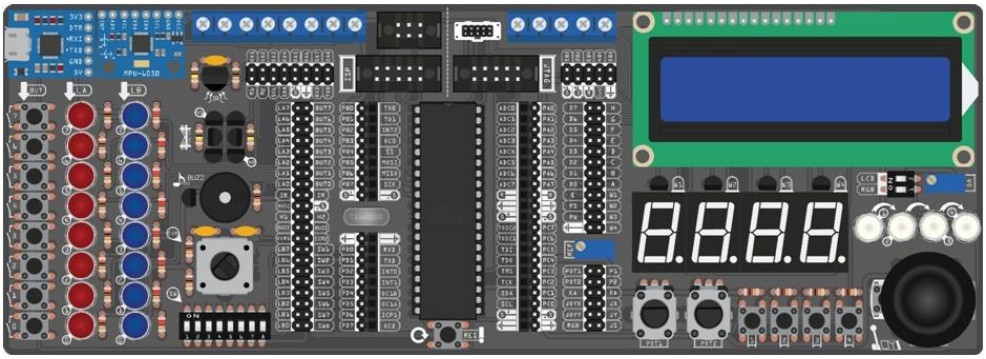

Fig. 1. Vector model of the startup system [Świtalski 2017]

For educational purposes, a graphical vector model of the board was prepared using the ILLUSTRATOR software (Fig. 1). It forms a numerical representation of a simplified startup system, with a representation of the geometry and the colours of all elements, which can be converted to a bitmap of any resolution. The graphical model provides a simple, intuitive way to show the connections between devices on the board or connected to it. The graphical diagrams minimise the likelihood of incorrect connection and are easy to understand for anyone, so that using them in popular science publications and courses expands the potential customer pool.

4. PCB DIAGRAM AND DESIGN

The diagram and design of the startup system's PCB were prepared using the EAGLE software – the diagram and design drawings form part of the documentation. Fragments of the diagram can be used during the educational process; therefore it was important to make them clear and unambiguous. For this reason, most default EAGLE symbols were replaced with our original markings, easily associated with the actual elements (Fig. 2). EAGLE element libraries were supplemented with additional symbols for the elements used in the design. Element outlines were also provided, so that the designer knows how the entire system will look after assembly. The EAGLE files are a valuable template for use in PCB design.

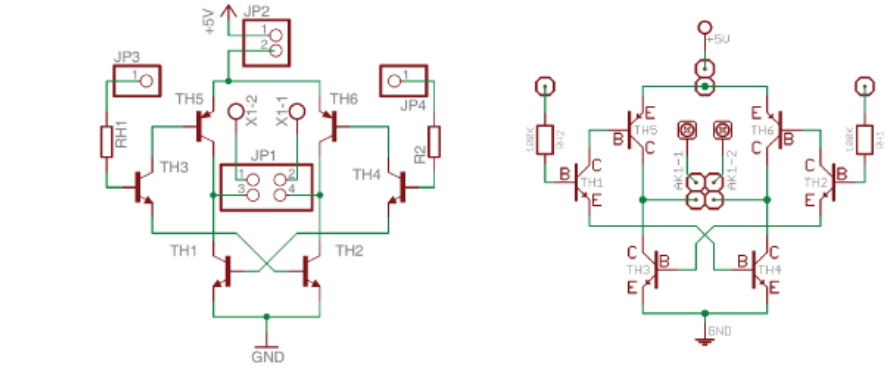

Fig. 2. Example of replacing EAGLE symbols with original symbols [Świtalski 2017]

The assumptions related to the arrangement of the microcontroller's connections with the peripherals in the central section of the board (Fig. 3) added complexity to the design, preventing traces from being routed only on one side. It became necessary to design two trace layers, TOP and BOTTOM. Most traces are 16 mil (approx. 0.41 mm) wide, with the smallest gaps of approx. 8 mil between the traces routed between the GOLDPIN strips. The 5 V line is routed twice, and has a width of 32 mil for most of its length. POLYGON is cast on both layers, filling the unused space with the GND potential, reducing the etching compound consumption during production and improving the board's electrical parameters. The PCBs were made on the basis of GERBER production files, generated using a CAM tool.

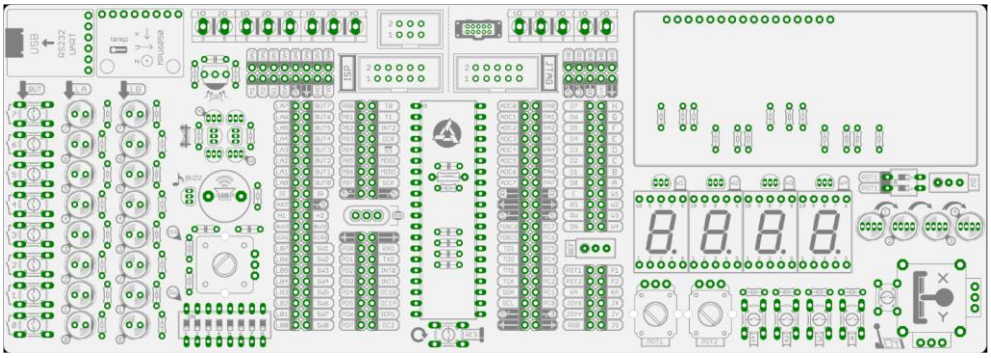

Fig. 3. Arrangement of elements in the EAGLE software [Świtalski 2017]

5. MICROCONTROLLER PROGRAMMING – ACHIEVING THE EDUCATIONAL GOALS

Entry requirements for students commencing microcontroller programming laboratory classes include a basic knowledge of programming in the C language, and general knowledge of microcontrollers, their internal organisation, general purpose input-outputs (GPIO), universal synchronous asynchronous receiver transmitters (USART), communication using Serial Peripheral Interface (SPI) buses, two-wire (TWI) and one-wire interfaces.

An educational principle was adopted that no ready-made libraries for buses or devices operating on these buses are used during the laboratory classes – all have to be written by the students, gradually creating their own library resource bases.

The laboratory exercise sequence includes writing a program for the microcontroller, selecting the lines for connecting elements or external devices, performing assembly in accordance with the design, and running the program in the evaluation system.

The exercises are grouped into two thematic blocks. The first block includes exercises focused on utilising the microcontroller's internal devices, i.e. timers, interrupt module, analogue digital converter (ADC). The exercises include configuring a GPIO line, digital control (e.g. diodes), reading line states (e.g. switch, rotary encoder), configuring timer operation, controlling devices using pulse width modulation (PWM) waveforms (e.g. RGB diode) and ADC configuration for reading analogue voltages (e.g. from a potentiometer, joystick, etc.).

The laboratory designs used in the second block of exercises focus on executing microcontroller communication with selected external devices (these include external EEPROM memory, digital potentiometer, real time clock, gyroscope-accelerator) and serial communication over a USART connection or over SPI, TWI or 1-wire buses.

Executing data transmission with a selected device on a bus requires the student to program the microcontroller interface registers and read/write in the external device.

After completing all exercises, the student possesses a set of library functions, written completely on their own and tested in the laboratory, useful in many projects involving AVR controllers.

Additional opportunities are provided by the availability of the startup board documentation, i.e. electrical diagrams, trace routing and element arrangement diagrams, and an original EAGLE element library. The documentation can be used in designing boards for individual student projects. New sensors can complement or replace those currently installed on the board, and this requires creating EAGLE models for them, taking into account their size and output layout. Performing

a project of this scope requires a financial contribution from the student and may form part of a diploma project.

6. SUMMARY

The startup system presented here constitutes part of the equipment for a Microprocessor and Embedded System Technology laboratory, one which has confirmed usefulness in teaching Atmel microcontroller programming in the C language and in the Amtel Studio environment. Together with a set of libraries facilitating the use of the implemented components, the startup system forms a useful test and educational laboratory platform for teaching and running original designs.

REFERENCES

AVR Butterfly Evaluation Kit User Guide, Atmel, 4271C-AVR-04/05, http://ww1.microchip.com. downloads/en/DeviceDoc/doc4271.pdf.

GrandEVBavr płyta ewaluacyjna dla mikrokontrolerów avr – instrukcja użytkownika, ver.1.00, 11/02/08, http://www.propox.com/download/docs/GrandEVBavr%20pl.pdf.

Kardaś M., 2016, *Instrukcja ATB-1.05A*, ATNEL 2016, wersja 1.4, https://www.atnel.pl/download/-elektronika/atb105/Instrukcja_ATB_1_05a.pdf.

MicroBoard for AVR 64-pin Manual, MicroElektronika, https://download.mikroe.com/documents/full-featured-boards/universal/unids-v6/unids-v6-mikroboard-avr-manual-v100.pdf.

Pałczyńska, B., Rabczuk, D., Fornalski, J., 2017, *Aplikacja monitorująca i sterująca systemem mikrokontrolerowym*, Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej, nr 9, Gdańsk.

Rabczuk, D., 2011, *Podsumowanie doświadczeń nauczania programowania mikrokontrolerów 8-bitowych AVR w środowisku AVR Studio*, Zeszyty Naukowe Akademii Morskiej w Gdyni, nr 70, Gdynia.

Świtalski, E., 2017, *Projekt Wykonanie testowo-dydaktycznej platformy laboratoryjnej z mikrokontrolerem 8-bitowym*, praca dyplomowa inżynierska, Akademia Morska w Gdyni, Wydział Elektryczny, Gdynia.