Development of Graphical User Interface for Individual Programming of Microcontrollers

Authors: M. Schmiedel¹, L. Szeliga², J. Wei³, A. Zimmermann⁴
¹Kantonsschule Sursee, ²Kantonsschule am Burggraben, St. Gallen, ³Calvin Genève, ⁴Calvin Genève

Supervised by: Lorenzo Ferretti

From 10 September to 16 September 2017

Abstract

Trying to program a microcontroller circuit-board can be very difficult for people without or with a little knowledge in writing the source codes in machine-near, not easy understandable to humans languages like C. Because of this, we have started to develop an intuitive visual interface allowing such people to realize even big projects without writing one single line of code. We adapted to and tested our graphical user interface on the Arduino Uno hardware. Already after several days of work we have showed that with our early-beta release application the simple projects can be accomplished.

1. Introduction

Microcontrollers have been popular since the 1980s. Without their existence, the life we know today could not be managed in an efficient way. They are used in so many different devices and tools, that one can say, that everything what modern human use, contains at least one or is influenced by a microcontroller. They have been created as a low cost, sufficient performance and functionality control devices with CPU and surrounding peripherals. They are often integrated on one single chip, smaller than a nail of a thumb, and can be inexpensively produced in high volumes. Because they are so small, they consume very little energy which is the resource in increasing demand in our civilization. Therefore, the microcontrollers are used where great computing power is not needed, where just to work on some input and output values is necessary. Thus, they can be found in television sets, phones, washing machines, fridges, coffee machines, baby phones, remote controls, etc.

The application spectrum of microcontrollers is very wide. Their diffusion for home-made and DIY projects has increased in the last 5 years. Therefore, in our research, we have concentrated on the hobby branch, where people need to program microcontrollers by themselves for the self-made projects, like e.g. hooking up a rain sensor, in order to send the needed input to a servo-motor for closing a window. There have been many start-up companies trying to support such applications with programmable microcontroller circuit-boards, where functionality can be defined by a
user and output on assigned ports can be made dependent on input received on other ports. Writing a software program for such circuit-boards, however, requires low-level languages (assembler, basic, C) programming skills. One of those start-ups called their microcontroller circuit-board Arduino and it can be coded in C. C, however, is a complicated, hard to learn, confusing and frustrating language when somebody just wants to make a simple Arduino-based project. Of course, there is a supported, simplified C version only for Arduinos and the community for projects is enormous, but there are no tools to create a source code directly and intuitively.

**Problem statement:**
How can we help people that do not know C, and are not interested in writing the C code by themselves, to create their projects in a simple way and get the microcontrollers work?

**Solution proposal:**
Make a simple and intuitive visual programming interface, a so-called GUI (Graphical User Interface), where the user can see what he is doing and can easily find bugs and resolve them.

**2. Materials & Methods**

For our project, we used Arduino HW with the corresponding programming and uploading environment, and Python IDE.

Arduino is open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. The hardware can interact with sensors and control lights, motors, and other actuators. The microcontroller on the board is programmable from Arduino Software (IDE) using the Arduino programming language, which in fact is a customized, simplified version of C. Arduino projects can be stand-alone or they can communicate with software running on a computer. There is a huge community for this platform which can help in case of any question.

![Arduino logo](image1.png)

**Figure 1:** Logo of Arduino

![Arduino microcontroller circuit-board](image2.png)

**Figure 2:** Example of Arduino microcontroller circuit-board
Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It is made very attractive for fast program development, as well as for using it to connect existing components together. Python is simple and easy to learn. It supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available for free for all major platforms in source and binary form and may be freely distributed. For our GUI, we had to use the TkInter library in order to make an interactive surface for the user.

Figure 3: Python logo

A basic construct of the GUI with built-in C generator has been proposed by our tutor. During initial brain storming sessions, we collected many ideas how to enhance and improve the GUI, which visual features add to the GUI and which control functions make initially available for a user. The ideas pool was changing until some of them have been finally implemented. We logically distributed within the team the different parts of the project in order to be more efficient.

Our work was concentrated mainly on the development of the GUI in Python IDE. We have been also writing some additional C code for the generator and optimizing C code generating. In Arduino Software (IDE) we have been debugging the generated C code, compiling it and uploading the final program to the microcontroller. Then we have been testing the functionality of the program on the circuit-board. The main challenge for us was to find a way to write a software in Python and to let it generate the correct C code, in our case for the microcontroller in the circuit-board of the Arduino. After we discovered how to do it, the real joy started. We expanded the software with every useful function we found on the Arduino website, for example the generating of sound or using a delay in a program, and we implemented also the decision-making behavior based on the If-Else statement.

3. Results

At the end of our venture our GUI is offering a user the possibility to create simple projects in an intuitive, visual way. On the initial screen, the variables can be assigned to any digital or analog pin (Fig. 4).
Then, all the states of a program can be setup and combined in the way that users want. Our GUI generates afterwards a diagram, which clearly shows to user the general structure of a program (Fig. 5).

Figure 4: Visual assignment of a variable to a pin

Figure 5: Visual presentation of state transition (Visual «code» of a blinking LED)
As next, with the pre-made blocks, a user can specify what the states actually will do (Fig. 6).

Finally, the “Generate Code” can be selected in the GUI and the C code for Arduino will be available (Fig. 7) for compilation and upload to the microcontroller.

After uploading of the compiled program to the microcontroller, the realized project can be seen in action (Fig. 8).
4. Discussion

During this study week, we showed that making the simple projects on the microcontrollers can be made available to people without low-level programming skills. We have created a functioning and clearly arranged, intuitive graphical user interface for the Arduino Uno microcontroller circuit-board. With our GUI, a user is able to create his own program without writing one single line of C code. This simple interface allows users of all levels to work on common projects. In this way, we reached our goal. It has to be mentioned, however, that working with our GUI still requires the basic knowledge of how microcontrollers work.

The current version of our software is an early-beta release and thus not all the features work properly. There are still many functions to be added (more build-in blocks) and improvements to be done (e.g. design). Also, the compilation of the C code and program uploading to the microcontroller directly from our application could be added.

Although in our program we have implemented the source code generator only for the Arduino Uno, it is possible to make it work for every kind of microcontrollers.

5. Acknowledgements

During this study week, we have learned to work on an informatics project as a team. We shared our ideas, distributed tasks and worked hard to realize them. Furthermore, we have learnt more about Python and C and improved our programming skills and experience.
We want to express our deep gratitude to everybody who enabled us to spend this fascinating and instructive week in Lugano, to do research in a very actual topic, in the ambience of the student’s life.
Very special thanks to our tutor Lorenzo Ferretti, who guided us through this project, always answered our questions, explained solutions for seemingly unsolvable problems and thus helped us out of black despair.
Without the support and funding from the HASLERSTIFTUNG this week could not take place. Thank you very much.

References

Where we shared our code: https://github.com/
Our second tutor: https://stackoverflow.com/
Python: https://www.python.org/
TkInter library: https://docs.python.org/2/library/tkinter.html
Arduino and its simplified C language: https://www.arduino.cc/