IIAI Open Conference Publication Series
Volume title
Volume number, Article ID
DOI: https://doi.org/10.52731/XXXXXX

Design and Implementation of Specialized Integrated Development Environment for IoT Programming Experiment Course

Yudai Kimbara * and Motoki Miura †

Abstract

The department to which the authors belong started an IoT (Internet of Things) programming experiment for third-year university students in 2021. This experiment involves the development of programs written in the C language, specifically tailored for the M5StickC-Plus platform. In the preceding years, the authors employed tools such as the Arduino IDE, the command-line build tool arduino-cli, and the Visual Studio Code Arduino extension to create their experimental environment. However, configuring these tools for the desired environment proved to be intricate, and managing serial connections during the coding process posed further complexities. To address these challenges, the authors designed a streamlined integrated development environment (IDE) using Java language. This dedicated IDE excels in presenting an organized array of sample source files, facilitating source code editing, and enabling efficient device programming. Consequently, the issues encountered while using the Arduino IDE and Visual Studio Code were successfully mitigated, resulting in a smoother progression of experiments for the students.

Keywords: Cross-compiling, microcomputer programming, arduino-cli, IoT education

1 Introduction

Our department has been offered a experimental course for third-year university students. As one of the five experimental themes, we have been offering a theme named "Internet of Things (IoT) Programming." In this experiment, students are expected to write programs in the C language to run on the *M5StickCPlus*, with a Project-based Learning (PBL) style [1]. The M5StickCPlus is a compact battery-powered microcontroller board that features a liquid crystal display (LCD) (see Figure 1) , Bluetooth/Wi-Fi capabilities, a red LED, an infrared LED, a microphone, a buzzer, acceleration/gyro sensors, and ports for connecting external electronic components and sensors (see Figure 2).

Students install a development environment on their own PCs (Windows, MacOS or Linux), form teams of four, conduct research, design, and build a feasible system while

* Chiba Institute of Technology, Chiba, Japan
† Chiba Institute of Technology, Chiba, Japan

communicating with the team members, and write a report. We believe that programming using a small multi-functional microcomputer with a LCD will stimulate students' interest and will be meaningful because they will be able to connect what they have learned in programming lectures to actual development.

Experimental lectures will be held for two weeks on five themes, and a presentation will be held at the end. Due to restrictions with other experimental themes, it is necessary to create deliverables, shoot a short video, and write a report in groups of four during two lectures (two weeks), four hours a week, and home studies. Therefore, we want to reduce the time and effort spent on building the environment as much as possible. In addition, considering the continuity with learning outside of class hours, we ask students to bring their own PCs that have completed the environment construction, if possible. Therefore, it is desirable that the work and setting procedures required for environment construction be kept to a minimum.

Figure 1: M5StickCPlus

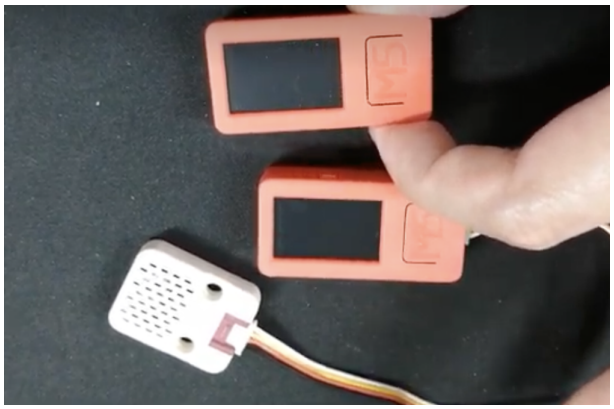

Figure 2: M5StickCPlus with temperature and humidity sensor

In 2021, we were experimenting with the Arduino IDE (version 1.8 at the time)[2]. The Arduino IDE can also be installed from the Microsoft Store. Therefore initial installation was easy. Also, it was convenient because the serial monitor (see Figure 3) is automatically disconnected when writing the program on the compact microcontroller board. However, in order to develop the M5StickCPlus on the Arduino IDE, it was necessary to configure settings while displaying multiple screens, such as boards, libraries, and ports to be used, as shown in Figure 4, had a high degree of difficulty for novices. Also, in order to avoid the garbled characters shown in Figure 5, it was necessary to set the line feed and baud

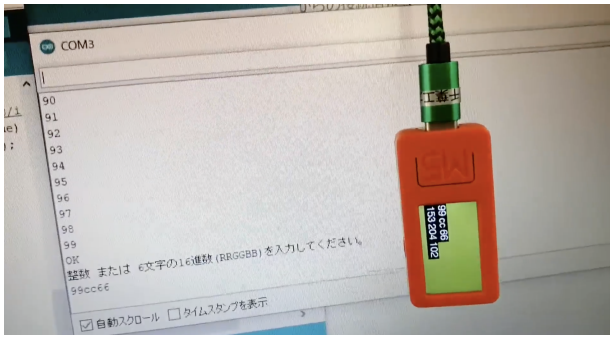

Figure 3: Serial monitor in Arduino IDE (M5StickCPlus received a message from the PC via an USB serial connection)

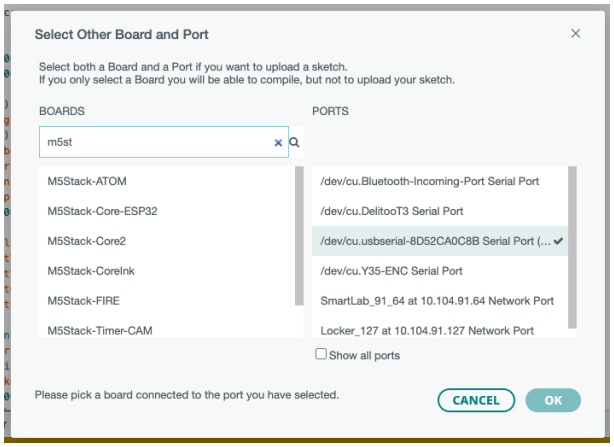

Figure 4: Configurations of Boards and Serial Ports in Arduino IDE

rate on the serial monitor. The 2022 experiment, we did not use the Arduino IDE, but rather the command-line version of the build and write tool *arduino-cli* [3]. The files can be edited using any editor (Visual Studio Code was recommended). The board setting and library introduction of arduino-cli were executed by using a shell script which we prepared in advance. We also prepared some shell scripts for compiling and writing for smooth procedure of experiments. However, the serial monitor of arduino-cli could not transmit to the device, and it was difficult to check the operation of the program. Also, the students had to manually disconnect the serial monitor when writing the program on the board. We also used the method of adding Arduino extensions to Visual Studio Code, but it was still complicated to configure such as the execution path.

2 Design and Implementation of *IoTP*

In this section, we describe the design and implementation of the integrated development environment specialized for our experimental course.

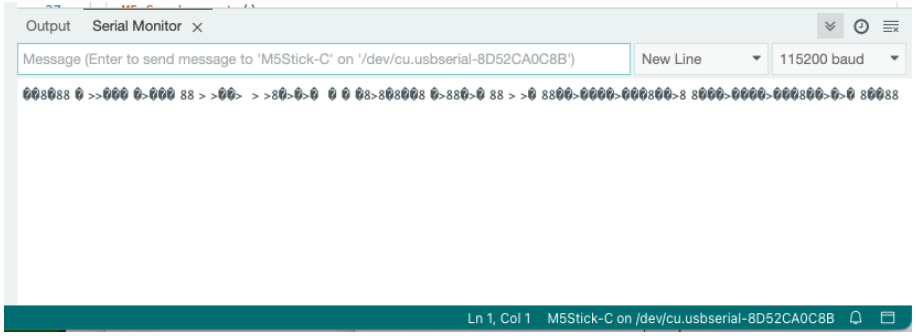

Figure 5: Garbled characters of serial monitor in Arduino IDE appears when a baud-rate is mismatched.

2.1 Design

IoTP is a development environment specialized for the development of M5StickCPlus with the following functions and features. We have developed the IoTP in Java language.

- When started, IoTP automatically searches for folders containing sample files provided by instructors. If more than one folder is found, the selection screen is displayed.
- IoTP displays a list of sample files. When a student selects a file, an editor is launched to edit that file.
- IoTP has the minimum necessary functions such as source code syntax highlighting, saving edited code, automatically adjusting indentation, search, and writing programs to the M5StickCPlus device.
- The student simply presses the upload button to start the build and write if the build is successful. If writing is successful, the serial monitor will be displayed. The baud rate of the sample file is preset to match the serial monitor settings. The serial monitor has a text input field for serial transmission, and the input string can be sent to the PC side.
- If the serial monitor is already open when you press the Upload button, IoTP will automatically disconnect in preparation for writing (uploading).
- IoTP also provides MQTT client function. The students can easily check MQTT transmission and reception in experiments.

In addition, we prepared a shell script that installs the arduino-cli command necessary for building, board settings, libraries, sample files provided by instructors, Java Runtime, etc. As a result, students can reduce the time and efforts involved in building an environment and focus on their experiments and training activities.

2.2 Installation and Startup

As previously mentioned, we prepared a shell script. The students can install IoTP by executing the following one-line command from a terminal such as Git for Windows[4]. By

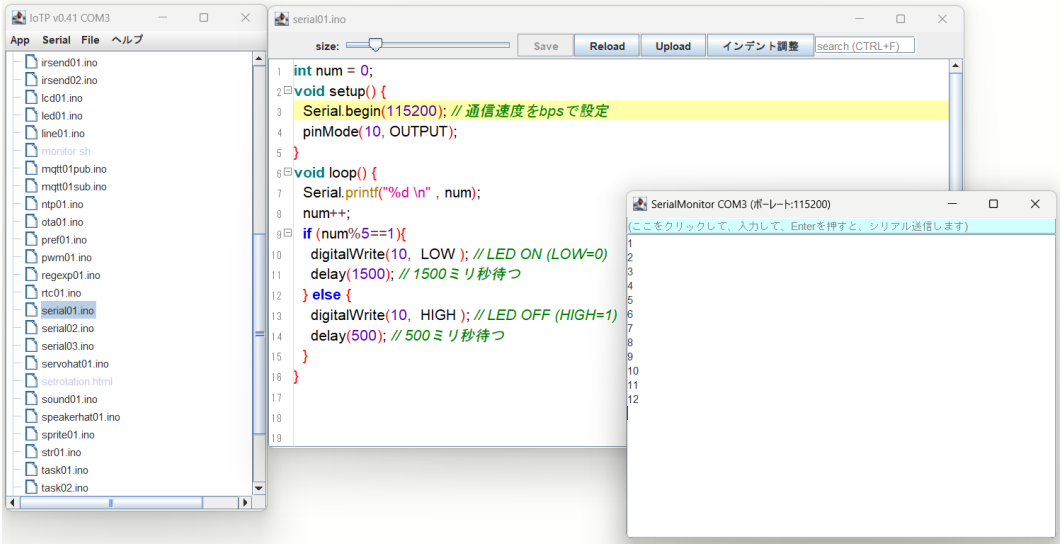

Figure 6: Screen of IoTP (Left) Sample File List (Center) Editor (Right) Serial Monitor Window

using Git for Windows, we can utilize commands such as git and curl, those are equivalent to MacOS and Linux.

```
$ curl -fsSL istlab.info/iotinst | sh
```

The students can directly run the IoTP JAR file to start IoTP, but the following shell script checks whether the IoTP version is up to date, downloads it if necessary, and then runs it.

```
$ curl -fsSL istlab.info/iotrun | sh
```

2.3 Usage

Figure 6 shows the screen of IoTP. The left is the file list window, the center is the editor, and the right is the serial monitor. Only the file list window is displayed at startup. When a M5StickCPlus is connected with a USB cable and the device can be accessed via the serial port, the serial port name is displayed in the title bar of the window. It also automatically sets the default serial port for uploading binary code as well as communicating. When a student double-clicks a sample file displayed in the file list window, the editor is launched as a single window (see Figure 6 center). The student can open multiple editors at the same time. The student can refer to the sample source codes displayed in the editors and make corrections as necessary. When the Upload button is pressed, save the source code, build it, and upload if the build is succeeded. After uploading, the serial monitor will be automatically displayed. If the student continues to modify the program and presses the Upload button, the connected serial monitor will automatically disconnect in preparation for the upload. This prevents the connection by the serial monitor from interfering with the connection for writing.

2.4 Implementation tricks and details

In the Arduino IDE [2], one program is basically stored in one folder as a project. However, when dealing with a large number of sample programs, there was a problem that the number of folders would increase if each was made into an individual project. Therefore, in IoTP, one program can be handled as one file (*.ino) for convenience. In fact, when the student presses the Upload button of an editor, IoTP copies the file that is open in that editor to the common project folder TestBuild with the file name TestBuild.ino. Then build the TestBuild folder. This makes it possible to easily refer to and edit programs without being aware of the project folder.

A sample program for experimentation is provided by Git [5], and IoTP also has a built-in Git client. In the case of PCs lent by the university instead of brought in, the user changes every two weeks, so the editing results of the previous person can be checked using the Git function, and can be restored to the initial state when necessary. The syntax highlighting and search functions of the editor use RSyntaxTextArea [6]. In addition, jSerialComm [7], a serial communication library, is used to implement a serial monitor in Java. The build uses the arduino-cli command.

3 Operations

From April 2023, the development environment IoTP is being used in actual experimental lectures. In order to explore the functions that can be realized with M5StickCPlus and how to implement them, students should refer to sample source code, build, write, run, perform serial communication, and check whether the running program works as intended.

Supports from TAs and instructors were required for the steps up to starting IoTP and identifying the port number corresponding to the device using the device manager when multiple serial ports were detected. But other situations there had been almost no problems or responses related to the development environment. Therefore, the students could focus on the design of the deliverables (systems) they want to realize and the activities to implement them with source codes.

3.1 Student Comments

We asked the students to write “impressions, opinions, improvement plans, etc.” at the end of the experimental lecture. The following is a part of the comments. In addition, there were some comments on lack of working time duration, but overall, no serious problems related to the integrated development environment were presented.

- It was a very good experience because I had never experienced the PBL format before. The preparation was completed with a single command line, and it was very easy and very good.
- Until now, in programming classes, we had only had one person create and execute the program, so it was refreshing and fun to create a program with multiple people like this time.
- It may be easy to do with Arduino IDE as long as you describe the procedure for building the environment. The source code for Wi-Fi connection was very helpful.
- I would like to know how to use the Arduino IDE in detail.

- There is a unique program IoTP, and it was very easy to use because it can be used with Git Bash. I was not familiar with the original program IoTP yet, so I didn't know where to edit or how to create a file, so I sometimes deleted one file and used it. I would like to improve on that.

- The text on how to prepare in advance was a little hard to read, so it turned out that it was not done on the day of the event, so I would appreciate it if you could make it a little easier to read.

- Although it is the original software used in this experiment, it is difficult to use because it is only compatible with M5StickCPlus. I felt it was better to use Arduino IDE even if the installation is little complicated and it requires some set-ups.

4 Related Research and Systems

Chedup et al. [8] compare runlinc IDE, a browser-based development environment for IoT STEM and AI education, with Arduino IDE. From the survey results, they conclude that runlinc IDE can easily and quickly realize IoT and AI application development. The runlinc IDE connects to a web server running on a device (STEMSEL Microcontroller) and is developed using HTML and Javascript. Several visual development environments have been proposed to lower the programming threshold. Pratomo et al. [9] builds a visual programming environment similar to Google Blockly for Arduino development. UIFlow [10] is a visual programming environment that runs on a web browser and supports devices such as M5StickCPlus and M5Stack. UIFlow adopts visual programming that combines blocks, so it is highly effective in lowering the threshold of programming. Since it is not written directly to the device but operates on the firmware, it is necessary to write the firmware in advance[1]. Therefore, development with IoTP and arduino-cli is superior in terms of the degree of freedom in development that utilizes the computational resources of the device.

5 Conclusions

In this paper, we introduced the Integrated IoT Programming Environment (IoTP) designed to facilitate the smooth progression of IoT programming experiments. By confining support to specific devices, the need for complex configurations was eliminated, thereby reducing the setup burden associated with environment preparation. Furthermore, effective management of serial monitor connections during program uploads contributed to an increased success rate in writing operations. Through these refinements, the constrained time allocated for experimental lectures could be optimally utilized, enabling seamless experimentation and exercises.

Although the introduced IoTP is tailored to the M5StickCPlus, an integrated development environment, its underlying principles can be readily adapted with minor modifications to accommodate microcontrollers and devices supported by arduino-cli.

IoTP has been released the source code at the following URL.
`https://git.istlab.info/miura250/IoTP`

[1] `https://docs.m5stack.com/en/quick_start/m5stickc_plus/uiflow`

Acknowledgments

The part of this research was supported by JSPS KAKENHI Grant-in-Aid for Scientific Research JP22K12319.

References

[1] Motoki Miura. IoT Programming (Experimental instructions) . https://cit.istlab.info/m5stickcplus/index.html. (Confirmed at August 20, 2023) (in Japanese).

[2] Arduino SA. Arduino IDE. https://www.arduino.cc/en/software. (Confirmed at August 20, 2023).

[3] The Arduino Team. Arduino CLI. https://github.com/arduino/arduino-cli/. (Confirmed at August 20, 2023).

[4] Johannes Schindelin, Lars Schneider, Edward Thomson, Matthew John Cheetham, and Matthias Aßhauer. Git for Windows. https://gitforwindows.org/, 2015. (Confirmed at August 20, 2023).

[5] Jon Loeliger and Matthew McCullough. *Version Control with Git: Powerful tools and techniques for collaborative software development*. O'Reilly Media, Inc., 2012.

[6] Fifesoft. RSyntaxTextArea – A Syntax Highlighting Text Component. https://bobbylight.github.io/RSyntaxTextArea/, 2015. (Confirmed at August 20, 2023).

[7] Fazecast Inc. jSerialComm – Platform-independent serial port access for Java. https://fazecast.github.io/jSerialComm/ (Confirmed at August 20, 2023).

[8] Sangay Chedup, Dushantha Nalin K. Jayakody, Bevek Subba, and Hassaan Hydher. Performance comparison of arduino ide and runlinc ide for promotion of iot stem ai in education process. In E. S. Gopi, editor, *Machine Learning, Deep Learning and Computational Intelligence for Wireless Communication*, pages 237–254, Singapore, 2021. Springer Singapore.

[9] Adin Baskoro Pratomo and Riza Satria Perdana. Arduviz, a visual programming ide for arduino. In *2017 International Conference on Data and Software Engineering (ICoDSE)*, pages 1–6. IEEE, 2017.

[10] M5Stack. UIFlow. https://flow.m5stack.com/. (Confirmed at August 20, 2023).