IoTプログラミング実験に特化した統合開発環境の設計と実装

金原 雄大[1,a)]　三浦 元喜[2,b)]

概要：著者らが所属する学科では，2021 年から M5StickC Plus 上で動作するプログラムを C 言語で開発する IoT プログラミング実験を 3 年次前期に開講している．昨年まで実験における開発環境として Arduino IDE や，コマンドライン版のビルドツールである arduino-cli，Visual Studio Code の Arduino 拡張などを使用していたが，環境構築における設定が複雑，書き込み時のシリアル接続管理が煩雑といった問題があった．そこで我々はサンプルソースファイルの一覧表示とソースコードの編集，デバイスへの書き込みに特化したシンプルな統合開発環境を Java プログラムとして構築した．これにより，Arduino IDE や Visual Studio Code を使用していたときの課題を解決し，学生は円滑に実験をすすめることができるようになった．

キーワード：クロスコンパイル，マイコンボード書込み，arduino-cli，IDE，演習環境

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

1. IoTプログラミング実験と既存開発環境の問題点

著者らが所属する学科では，2021 年から M5StickC Plus 上で動作するプログラムを C 言語で開発する IoT プログラミング実験を 3 年次前期に開講している [1]．M5StickC Plus は，液晶ディスプレイと Bluetooth/Wi-Fi，赤色 LED，赤外 LED，マイク，ブザー，加速度/ジャイロセンサ，外部の電子部品・センサとの接続用ポート等を備えた，バッテリー駆動可能な小型マイコンボードである．

学生は自身の PC（Windows, MacOS または Linux）に開発環境をインストールし，4 人 1 組の班となり，班員とコミュニケーションをとりながら実現可能なシステムの調査と設計，構築を行い，報告書としてまとめる．液晶ディスプレイを備えた小型多機能マイコンを用いたプログラミングは学生の興味を促進するとともに，これまでのプログラミング講義で学んだ内容を実際の開発に結びつけることができるため，有意義であると考えている．

実験講義は 5 つのテーマについて 2 週ずつ実施し，最後に発表会を行う．他の実験テーマとの制約から，1 週 4 時間 ×2 回（2 週）の講義時間および自宅学習で，4 人 1 組の班による成果物の作成と動画撮影，レポート執筆を行う必要がある．そのため，環境構築にかける時間や労力はなるべく減らしたい．また学生には，時間外学習との連続性を考慮し，可能であれば環境構築を済ませた持ち込み可能な PC を用意してくるようにと伝えている．そのためにも，環境構築に必要な作業や設定の手順は少ないことが望ましい．

2021 年は Arduino IDE（当時はバージョン 1.8）[2] を用いて実験を行っていた．Arduino IDE は Microsoft Store からもインストール可能であるため，最初の導入は容易である．またプログラムを書き込むときにシリアルモニタを自動的に切断するため利便性が高い．しかし，M5StickC Plus の開発を行うためには，**図 1** に示すようなボードの

[1] 千葉工業大学 工学研究科 情報通信システム工学専攻
Chiba Institute of Technology, Narashino, Chiba 275–0016, Japan
[2] 千葉工業大学 工学部 情報通信システム工学科
Chiba Institute of Technology, Narashino, Chiba 275–0016, Japan
a) s19a5049er@s.chibakoudai.jp
b) motoki.miura@p.chibakoudai.jp

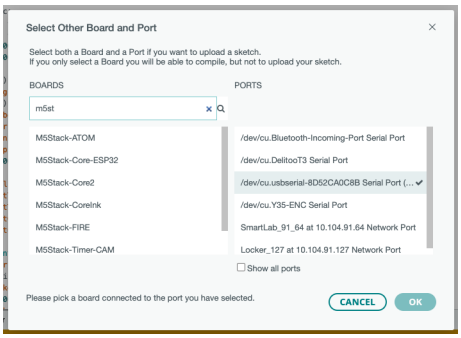

図 1　Arduino IDE におけるボードとポートの設定画面

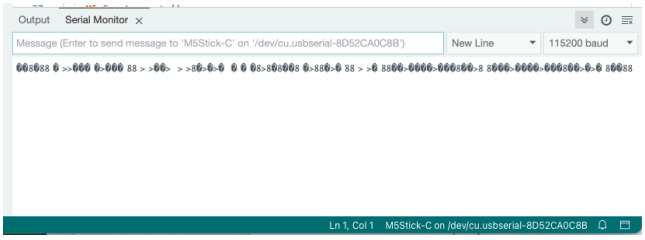

図 2　Arduino IDE シリアルモニタの文字化け

設定やライブラリの準備，ポートの設定など，複数の画面を表示しながらの準備が必要であり，初学者には難易度が高かった．また，シリアルモニタも**図 2** に示すような文字化けを回避するには改行設定やボーレート設定を行う必要がある．

2022 年の実験では Arduino IDE を使用せず，コマンドライン版のビルド・書き込みツールである arduino-cli [3] を使用した．ファイルの編集は任意のエディタ（推奨は Visual Studio Code）を使用してもらった．arduino-cli のボード設定やライブラリ導入は，あらかじめ用意したシェルスクリプトで行うことで省力化できた．また円滑な実験のためコンパイルや書き込み用のシェルスクリプトも用意した．しかし，arduino-cli のシリアルモニタではデバイスへの送信ができず，プログラムの動作確認が行いにくいことが問題であった．またプログラムを書き込むときにはシリアルモニタの接続を手動で切っておく必要があった．Visual Studio Code に Arduino 拡張機能を追加する方法も併用したが，こちらも実行パスの設定などが煩雑であった．

そこで我々は，Arduino IDE が提供していたシリアルモニタ管理の長所を活かしつつ，設定の煩雑さを解消するため，IoT プログラミング実験において使用する M5StickC Plus の開発に特化した統合開発環境 IoTP を開発した．

2.　統合開発環境 IoTP の設計と実装

ここでは，我々が構築した統合開発環境 IoTP の設計と実装について述べる．

2.1　設計

IoTP は，以下の機能および特徴を備えた，M5StickC Plus の開発に特化した開発環境であり，Java 言語で開発している．

- 起動すると，教員が提供するサンプルファイルを含むフォルダを自動的に探索する．複数見つかった場合は選択画面を表示する．
- サンプルファイルの一覧をリストで表示する．学生がファイルを選ぶとそのファイルを編集するためのエディタが起動する．
- エディタでは必要最低限の機能として，ソースコードのシンタックスハイライト機能，編集したコードを保存する機能，インデントを自動調整する機能，検索機能，プログラムを M5StickC Plus デバイスに書き込む機能を提供する．
- 編集結果を確認しやすくするため，Upload ボタンを押すだけで，ビルドを開始できるようにする．ビルドに成功したら書き込みを行う．書き込みに成功したらシリアルモニタを表示する．シリアルモニタにはあらかじめサンプルファイルのボーレートを設定する．シリアルモニタにはシリアル送信用のテキスト入力フィールドが備えてあり，PC 側からの送信ができるようにする．
- Upload ボタンを押したとき，すでにシリアルモニタが開いていたら，書き込み（アップロード）に備えて自動的に切断する．
- MQTT クライアント機能を備えており，実験における動作確認が行える．

なおビルドに必要な arduino-cli コマンドのインストールやボードの設定，ライブラリ，教員が提供するサンプルファイル，IoTP 自体の実行プログラムおよび Java Runtime 等，必要なものはすべてシェルスクリプトで導入できるようにする．これにより，学生は環境構築にかかる手間とトラブルを軽減でき，実験・実習活動に注力することができる．

2.2　導入および起動

環境構築は基本的に，Git for Windows[4] 等の端末から以下のシェルスクリプト

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

を実行することによって行っている．Git for Windows を用いることで，MacOS や Linux と同等のコマンドが利用できる．

IoTP の起動については IoTP の JAR ファイルを直接実行してもらってもよいが，バージョンアップに対応するため，基本的には起動のみ行う以下のシェルスクリプト

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

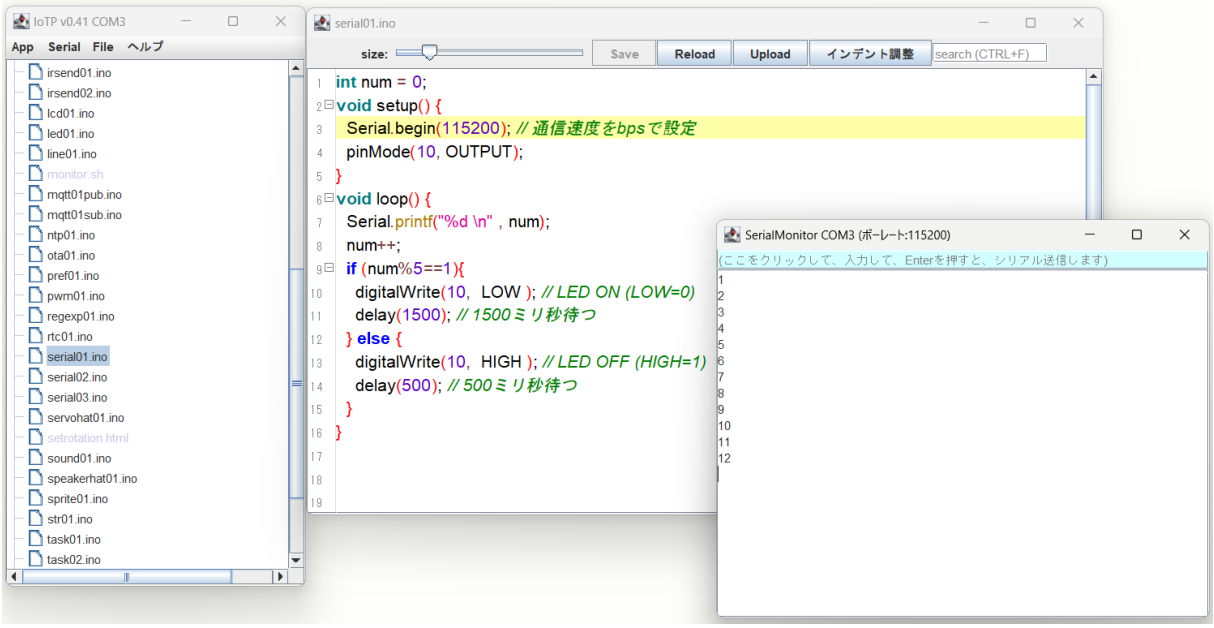

図 3 IoTP の画面 (左) サンプルファイルリスト (中央) エディタ (右) シリアルモニタ

を端末で実行してもらう．スクリプト内で最新のバージョンがダウンロードされているかを確認し，されていればそのまま起動，されていなければダウンロードして起動するようにしている．

2.3 利用

図 3 に，IoTP の画面を示す．左がファイルリストウィンドウ，中央がエディタ，右がシリアルモニタである．起動時はファイルリストウィンドウのみ表示している．M5StickC Plus を USB ケーブルで接続し，シリアルポート経由でデバイスにアクセスできるようになると，ウィンドウのタイトルバーに，シリアルポート名を表示する．また，自動的に書き込み用のデフォルトのシリアルポートが設定される．

学生がファイルリストウィンドウに表示された，サンプルファイルをダブルクリックすると，エディタが一つのウィンドウとして起動する．複数のエディタを同時に開くこともできる．学生はエディタに表示されるサンプルファイルを参照し，必要に応じて修正を行う．その後 Upload ボタンを押すと，プログラムを保存し，ビルド，アップロードを行う．アップロードに成功すると，シリアルモニタを自動的に表示する．引き続きプログラムを修正し，Upload ボタンを押すと，接続しているシリアルモニタはアップロードに備えて自動的に切断される．これにより，シリアルモニタによる接続と，書き込みにおける接続が干渉することを防いでいる．

2.4 実装上の工夫

Arduino IDE では，基本的に一つのプログラムはプロジェクトとして一つのフォルダに収めることになっている．しかし，多数のサンプルプログラムを扱う場合，それぞれ個別のプロジェクトにするとフォルダ数が多くなってしまう問題があった．そのため IoTP では，便宜上一つのプログラムは一つのファイル (*.ino) として扱えるようにした．実際には，エディタの Upload ボタンを押したとき，そのエディタで開いているファイルを共通のプロジェクトフォルダ TestBuild に TestBuild.ino というファイル名でコピーする．そして，TestBuild フォルダに対してビルドと書き込みを行う．これにより，通常はプロジェクトフォルダを意識せずに手軽にプログラムを参照したり編集したりすることができる．

実験用のサンプルプログラムは Git [5] で提供しており，IoTP にも Git クライアントが組み込まれている．持ち込みではなく大学が貸し出す PC においては，2 週間ごとに使用者が交代するため，前の人の編集結果を Git の機能によってチェックし，必要なときに初期状態に戻せるようにしている．

エディタのシンタックスハイライトや検索機能は RSyntaxTextArea [6] を使用している．また Java におけるシリアルモニタを実現するため，シリアル通信ライブラリである jSerialComm [7] を使用している．

3. 運用

2023 年の 4 月から，開発環境 IoTP を実際の実験講義で使用している．学生には，M5StickC Plus で実現できる機能と，その実装方法を調べるために，サンプルソースコードを参照したり，ビルド・書き込み・実行したり，実行中のプログラムが意図通り動作しているかをシリアル通信で確認したりする作業が求められる．IoTP を起動するまで

の段階と，複数のシリアルポートが検出されたときにデバイスに対応するポート番号をデバイスマネージャを用いて特定する状況では TA や教員のサポートを必要としたが，それ以外の場面で開発環境に関する問題や対応はほとんど発生していない．そのため，学生は実現したい成果物（システム）の機能と，それをソースコードによって実装する活動に注力できている．

4. 関連研究およびシステム

Chedup ら [8] は，IoT STEM 教育および AI 教育を目的としたブラウザベースの開発環境 runlinc IDE と，Arduino IDE の比較を行っている．調査結果から，runlinc IDE は IoT 開発や AI アプリケーション開発を簡単かつ迅速に実現できると結論づけている．runlinc IDE はデバイス（STEMSEL Microcontroller）上で動作する Web サーバに接続し，HTML や Javascript を用いて開発する形式であるため，本研究における開発環境とは目的および対象者が異なる．

プログラミングの敷居を下げるため，ビジュアルな開発環境はいくつか提案されている．Pratomo ら [9] は，Arduino 開発において Google Blockly に似たビジュアルなプログラミング環境を構築している．また UIFlow[10] は，Web ブラウザで動作するビジュアルプログラミング環境 であり，M5StickC Plus や M5Stack 等のデバイスに対応している．UIFlow はブロックを組み合わせる方式のビジュアルプログラミングを採用しているため，プログラミングの敷居を下げる効果は高が，デバイスに直接書き込むのではなく，ファームウェア上で動作する[*1] ため，あらかじめファームウェアを書き込んでおく必要がある．そのためデバイスの計算リソースを活用する開発の自由度という点においては IoTP および arduino-cli による開発のほうが優れている．

5. まとめ

IoT プログラミング実験を円滑にすすめるための統合開発環境 IoTP について紹介した．対応デバイスを限定したことにより設定が不要となり，環境構築における設定の負荷を下げることができた．またプログラムのアップロード時にシリアルモニタの接続を管理することにより，書き込みの成功率を高めることができた．これらの工夫により，限られた実験講義時間を有効に活用し，円滑な実験・演習が行えるようになった．

本稿で紹介した IoTP は M5StickC Plus に特化した統合開発環境であるが，基本的には arduino-cli がサポートするマイコンやデバイスであれば少しの改良で対応可能である．

IoTP は以下のサイトでソースコードを公開している．
`https://git.istlab.info/miura250/IoTP`

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

謝辞 本研究の一部は JSPS 科研費 JP22K12319 の支援によるものです．

参考文献

[1] 三浦元喜：IoT プログラミング （実験指示書），`https://cit.istlab.info/m5stickcplus/index.html`（2023 年 6 月 14 日確認）．

[2] Arduino SA: Arduino IDE, `https://www.arduino.cc/en/software`（2023 年 6 月 6 日確認）．

[3] The Arduino Team: Arduino CLI, `https://github.com/arduino/arduino-cli/`（2023 年 6 月 6 日確認）．

[4] Schindelin, J., Schneider, L., Thomson, E., Cheetham, M. J. and hauer, M. A.: Git for Windows, `https://gitforwindows.org/`（2023 年 6 月 6 日確認）(2015).

[5] Loeliger, J. and McCullough, M.: *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/`（2023 年 6 月 6 日確認）(2015).

[7] Fazecast Inc.: jSerialComm – Platform-independent serial port access for Java, `https://fazecast.github.io/jSerialComm/`（2023 年 6 月 6 日確認）．

[8] Chedup, S., Jayakody, D. N. K., Subba, B. and Hydher, H.: Performance Comparison of Arduino IDE and Runlinc IDE for Promotion of IoT STEM AI in Education Process, *Machine Learning, Deep Learning and Computational Intelligence for Wireless Communication* (Gopi, E. S., ed.), Singapore, Springer Singapore, pp. 237–254 (2021).

[9] Pratomo, A. B. and Perdana, R. S.: Arduviz, a visual programming IDE for arduino, *2017 International Conference on Data and Software Engineering (ICoDSE)*, IEEE, pp. 1–6 (2017).

[10] M5Stack: UIFlow, `https://flow.m5stack.com/`（2023 年 6 月 12 日確認）．