

# モジュール性の高いツールキットを用いたハードウェアプロトタイピングにおける初学者の課題の理解

伊東 大耀<sup>1,a)</sup> 篠田 和宏<sup>1</sup> 矢谷 浩司<sup>1</sup>

**概要:** ハードウェアプロトタイピングでは、プログラミングと電子回路というソフトウェア・ハードウェアの両方の知識を組み合わせるシステムを設計・実装する必要があるため、初学者は様々な困難に直面する。先行研究では、Arduino 等のモジュール性の低いマイコンを使用したハードウェアプログラミングにおいて初学者が直面する困難に関する調査が行われている。一方で、近年では、micro:bit 等のモジュール性の高いツールキットが開発され、これらのミスの多くはモジュール性の高いツールキットを使用することで防ぐことができると考えられる。そこで、本研究では、モジュール性の高いツールキットを使用したハードウェアプロトタイピングワークショップにおいて初学者が直面する困難を明らかにし、先行研究の結果と比較することでツールキットのモジュール性による初学者が直面する困難の違いを追究する。

## Investigating Difficulties for Novices in Hardware Prototyping with High-Modularity Toolkits

ITO TAIYO<sup>1,a)</sup> SHINODA KAZUHIRO<sup>1</sup> YATANI KOJI<sup>1</sup>

### 1. 序論

ハードウェアプロトタイピングとは、加速度・温度・光等のセンサや、ライト・ディスプレイ・モータ等の出力デバイスを、マイクロコントローラ（以下、マイコン）と繋いでシステムをつくることを通して、プログラミングや電子工作の技術を習得したり、アイデアの仮説検証を行ったりすることを指す。ハードウェアプロトタイピングは教育現場で普及が進んでおり、その目的としては、コンピュータサイエンスやものづくりに積極的に触れる機会を提供することで、ものづくりに興味を持ってもらったり主体的な学びに繋げてもらったりすることが挙げられる [10,11]。

このように、教育現場でハードウェアプロトタイピングが行われるにあたり、先行研究では、Arduino 等のマイコンを用いたハードウェアプロトタイピングにおいて初学者が直面する困難に関する調査が行われている [2,3,5]。そ

の結果、回路の配線や電子部品、ブレッドボード、プログラムに関する困難に直面するという知見が得られたが、これらのミスの多くはモジュール性の低いマイコンを使用していることが原因だと考えられる。ここで、モジュール性とは、独立して組み合わせることができる自己完結型のユニット（モジュール）に構成要素がカプセル化されていることを指す [1]。従来教育現場で使われていたモジュール性の低いマイコンでは、たくさんの配線やピン、電子部品の極性を考慮する必要があるのに対し、近年教育現場で使われているモジュール性の高いツールキットでは、基本的なセンサ・出力が内蔵されているためブレッドボードが不要で、また、内蔵されていないセンサ・出力を使う場合も規格化された端子でミスなく配線を行うことができる。このように、先行研究ではモジュール性の低いマイコンを対象に調査が行われているのに対して、現状の教育現場ではモジュール性の高いツールキットが普及している。つまり、先行研究では現状の教育現場に即した調査が行われていない。

そこで、本研究では、モジュール性の高いツールキット

<sup>1</sup> 東京大学 Interactive Intelligent Systems Laboratory  
IIS-Lab, The University of Tokyo, Bunkyo, Tokyo 113-8656,  
Japan

<sup>a)</sup> taiyo@iis-lab.org

として micro:bit<sup>\*1</sup>を用いたハードウェアプロトタイピングワークショップを実施した。本ワークショップは電子工作の初学者 15 人を対象とし、制作するデバイスについてアイデアを出し実装に落とし込むまでの一通りのものづくりの流れを行ってもらった。本ワークショップを通じて初学者が直面する困難を調査し、さらに、モジュール性の低いマイコンを対象とした先行研究の結果と比較することで、初学者が直面する困難にツールキットのモジュール性がどのような影響を与えるのかについても明らかにした。最後に、これらの観点を踏まえてモジュール性の高いツールキットを用いたハードウェアプロトタイピングにおいて初学者をどのように支援できるのかについて検討した。

## 2. 関連研究

### 2.1 ハードウェアプロトタイピングと教育

2016 年に、英国放送協会 (British Broadcasting Corporation, BBC) によって教育用マイコンボード micro:bit がイギリスの児童 100 万人に無償配布された<sup>\*2</sup>。配布された micro:bit がどのように使われているかを調査した研究 [10, 11] によると、多くの生徒が高い熱意や興奮を抱いており、コンピュータの画面上だけでなく現実世界で実際に自分で制作したものを目にするができるため、プログラミングに取り組む意欲が向上することや、何を作るかというアイデアの部分で創造性を発揮できるため、主体的な学びや矜持に繋がることがわかった [4, 8, 9]。

また、Kafai ら [6] は、高校のコンピュータサイエンスの授業において、LilyPad Arduino を使った e-テキスタイル制作を行い、e-テキスタイルの高校カリキュラムへの導入の有効性を示した。ここで、e-テキスタイルとは、センシング機能を持たせた繊維製品のことを指す。生徒たちは、少なくとも 4 つの LED と 2 つの導電性パッチを使用するという条件の下、自由に e-テキスタイルを設計・実装した。この先行研究では、このように部分的に設計要件を定めたことで、生徒の学習と創造的な表現の双方を可能にすることができたとしている [10]。

### 2.2 電子工作の初学者が直面する困難

DesPortes ら [3] は、Arduino を使ったことがない生徒を対象に、Arduino を使って電子工作をするタスクに取り組んでもらい、その際に直面した困難を追究した。タスクの内容は、2 つの LED を使った回路をブレッドボードを用いて作成し点滅させるというものであった。ソフトウェアに関する困難としては、コードが実行される順序に関するミスが最も多かった。これは、例えば、無限ループの後にさらにコードを続けてしまったり、無限ループの中に別の無限ループを入れ子にしてしまったりするということであ

る。Arduino にはピンと呼ばれる接続端子が複数あるが、ピンに関するミスも多く見られた。例えば、コードにおいて誤ったピンを参照してしまったり、各ピンを入力・出力のいずれかで用いるのかを指定し忘れていたりしていた。ハードウェアに関する困難としては、回路を開放させてしまうというミスが最も多く見られた。これは、例えば、ブレッドボードに接続すべき 2 つの電子部品を差し込む際に、隣り合っているが電氣的に接続されていない 2 つの穴を利用してしまいうということである。他にも、直列・並列といった回路構成を誤ったり、LED を逆向きに接続したり、Arduino の誤ったピンに接続したりするといったミスが見られた。

また、Booth ら [2] は、Arduino を使ったことがある成人の非専門家を対象に、Arduino を使って電子工作をするタスクに取り組んでもらい、その際に直面した困難を分析した。ただし、タスクの内容は、温度センサの測定値を LED で可視化するというものであり、測定値が低ければ 1 つの LED が、中程度であれば 2 つの LED が、高ければ 3 つの LED が点灯するようにした。結果としては、Arduino の使用経験があるにも拘らず、タスクを完遂した人は 20 人中 6 人であった。また、直面した困難の多くは、プログラムに関連していたが、そのほとんどは時間内に修正できていた。一方で、タスク失敗に繋がった致命的な困難の多くは回路に関連していた。具体的には、温度センサや LED を Arduino の誤ったピンに接続するといった配線における欠陥が最も多く見られた。

さらに、Sadler ら [5] が行った研究によると、ステップバイステップの指示に従いながら Arduino やブレッドボードを用いて 10 分以内に 1 つの LED を点滅させるという基本的なタスクを高校生に課したが、タスク完了に人間の介入を必要とするほどの重大な困難に直面した参加者の割合は 47%にまで上った。

このように、電子工作の初学者が直面する困難を追究する研究は為されているが、これらは Arduino のようなモジュール性の低いマイコンおよびブレッドボードを使用している。しかし、Sadler ら [5] の研究によれば、電子工作における電子部品のモジュール化はエラーの確率を減少させるという。よって、micro:bit を中心としたモジュール性の高いツールキットを使用する場合、ピンや配線に関するミスは少なくなり、初学者が直面する困難は別にあると考えられる。

## 3. データの収集

モジュール性の高いツールキットを用いたハードウェアプロトタイピングにおいて初学者が直面する困難を調査するために、micro:bit を用いたハードウェアプロトタイピングワークショップを実施し、参加者の言動を記録した。

<sup>\*1</sup> <https://microbit.org/get-started/what-is-the-microbit/>

<sup>\*2</sup> <https://microbit.org/about/>

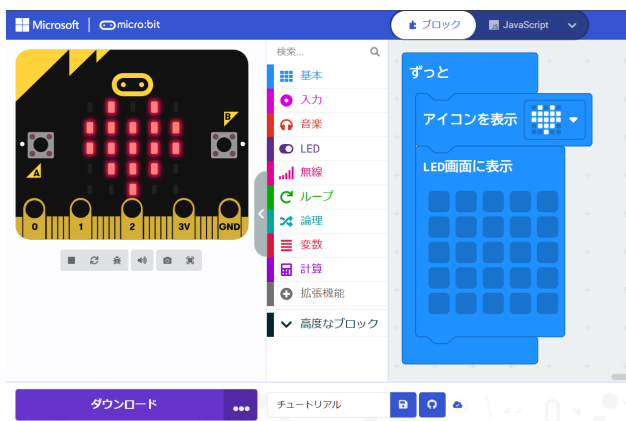


図 1: micro:bit 用の MakeCode エディタ



図 2: ワークショップにおける講義の様子

### 3.1 ワークショップの参加者

電子工作の経験がない成人を対象に、主に実験参加者募集サイトを通じて募集した。応募時に電子工作の経験について回答してもらい、その内容をもとに適切だと判断した15人(男9人、女6人)の希望者に参加してもらった。

また、Teaching Assistant (TA) を5人設け、ワークショップ中に実験参加者が困難に直面した際に質問を受ける役割を担った。TAは全員 micro:bit の使用経験がある本学所属の学生である。

### 3.2 ワークショップの構成

ワークショップは2日間実施され、その内容については、Katterfeldt らの研究 [7] を参考にして次の通りとする。

1) チュートリアル: micro:bit の使い方についてのチュートリアルを行い、2) 講義: プログラミングや電子工作の基礎知識に関する講義を行い、3) アイデア出し: 日常の不満や身近なテーマをもとにアイデアを考え、4) 実装: 自らのアイデアを実装し、5) 発表: 完成した作品について他の参加者の前で発表してもらった。

本ワークショップでは、参加者15人を3人ずつの計5チーム(チーム1、チーム2、…、チーム5とする)に振り分けた。その際、男女比およびプログラミング・電子工作の経験についてチーム間でできるだけ偏りが生じないようにした。ワークショップにおける参加者への説明(チュートリアルや講義等)は主に筆者がスライドをプロジェクタで映しながら行った。また、micro:bit のプログラムを書くためのオンラインプラットフォームとして MakeCode<sup>\*3</sup> を利用した。さらに、参加者一人ひとりに対し Grove Inventor Kit for micro:bit (以下、Grove Kit) を用意した。Grove Kit とは、micro:bit と接続することで様々なモジュールを利用することができる外付け部品セットであり、4種類の出力(4桁ディスプレイ、スピーカー、赤色 LED、LED テープ)と4種類の入力(ジェスチャーセンサ、回転角度

センサ、明るさセンサ、超音波距離センサ)からなる。ただし、スピーカーについては、micro:bit がバージョンアップしたことで micro:bit 本体にも内蔵されている機能であるため、本ワークショップの講義では扱わないものとした。

参加者が困難に直面した際には、1) Google Chrome を用いたオンライン検索、2) 生成 AI の Chatbot システムの利用、3) チームメンバーと議論、4) TA に質問、の4つの解決方法を利用してよいものとした。ただし、生成 AI の Chatbot システムについては研究者側で用意したもので、ファイルの入出力等には対応しておらず、テキストベースの入出力のみが可能で、OpenAI が発行する GPT モデルの API キーを用いて ChatGPT の UI による操作を可能にする Chatbot UI<sup>\*4</sup> を利用した。

#### 3.2.1 チュートリアル

micro:bit を使ってプログラミングを行うことに慣れてもらうためのチュートリアルとして、図1のように micro:bit の LED 画面にハートのアイコンを点滅させるという内容を扱った。micro:bit とパソコンの接続、MakeCode エディタの利用、プログラムの作成、micro:bit へのプログラムの書き込みについて、スライドで示しつつ実際に行いながら説明した。

#### 3.2.2 講義

本ワークショップにおいて行った「講義」は1) プログラミング入門、2) micro:bit でできること、3) Grove Kit、の3つのパートで構成されている。

どのパートにおいても図2のように、講義スライドで示したサンプルコードを参加者自身のパソコンで作成し、作成したプログラムを手元の micro:bit に書き込んでもらった。サンプルコードの作成および演習の解答例の提示については、筆者自身もその場で行いプロジェクタでその様子を示しながら説明した。

「プログラミング入門」パートでは、5つの基礎的な項目(四則演算、文字列、変数、条件分岐、繰り返し)を扱った。

<sup>\*3</sup> <https://makecode.microbit.org/>

<sup>\*4</sup> <https://www.chatbotui.com/ja>

「micro:bit できること」パートでは、「プログラミング入門」パートで既に利用したものも含めて、micro:bit に内蔵されている主な出力 2 種類（光らせる、音を鳴らす）と入力 7 種類（ボタンを押す、振る、傾ける、方角センサ、明るさセンサ、加速度センサ、温度センサ）と乱数を扱った。「Grove Kit」パートでは、Grove Kit に含まれるモジュールである出力 3 種類（4 桁ディスプレイ、赤色 LED、LED テープ）と入力 4 種類（ジェスチャーセンサ、回転角度センサ、明るさセンサ、超音波距離センサ）を扱った。

### 3.2.3 アイデア出し

本ワークショップにおいて行った「アイデア出し」では、たくさんのアイデアを出してから（発散）、それをもとにアイデアを厳選していく（収束）という流れでアイデア出しを行った。まず、各参加者にたくさんのアイデアを出してもらうために Crazy 8 という方法を利用し、厳しい制限時間を設けたうえで日常生活における不満を 8 つ紙に書いてもらった。次に、チームごとに 3 人分で合計 24 個の不満の中から今回解決を試みる不満を選んでもらうために Dot Voting という方法を利用し、チームごとに時計回りに各参加者の不満が書かれた紙を回しながら、賛成する不満にシールを貼ってもらい、シールの数をもとにチームごとに取り組む問題を決めてもらった。

このようにして決まった取り組む問題に対し、その原因を次々と挙げてもらい、各原因に対応する解決策も考え、チームごとに思考展開図としてまとめてもらった。思考展開図をもとに、micro:bit を使って解決できるかどうか、1 日という時間内で作れそうか等の判断基準で最終的な解決策をチームごとに決めてもらった。

### 3.2.4 実装

4 時間の制作時間を設けてチームごとに 1 日目にまとめてもらったアイデアをもとに自分に割り当てられたタスクを実装しチームで一つの作品を制作してもらった。なお、本ワークショップでは、講義資料を参加者に共有したり参加者が演習を終えることができたかを確認したりするために Slack を利用していたため、各チームのチャンネルを作成しチーム内でプログラム等の共有を行えるようにした。

### 3.2.5 発表

チームごとの実装終了後、他の参加者の前で制作した作品を発表してもらった。その際、取り組んだ問題とその解決策、仕組み等をまとめたスライドを使いながら、作品のデモンストレーションを行ってもらった。デモンストレーションについては、Zoom で画面共有しているスマホを使って他の参加者にも見やすいようにした。

## 3.3 困難の収集方法

Sadler らの研究を参考にして、「参加者が行き詰まった状態から抜け出すために他者の介入を必要とするイベント」

を 1 つの困難と定義する [5]。本ワークショップにおいて参加者が行き詰まった状態から抜け出すための 4 つの解決方法に対応して、本研究において収集したデータは、Google Chrome の検索履歴、Chatbot システムのログデータ、チームメンバーや TA とのやり取りを記録した録画データの三つである。検索履歴については、Google Chrome の拡張機能 History Plus<sup>\*5</sup>を利用して、2 日間の検索履歴を提出してもらった。生成 AI の Chatbot システムのログデータについても、システム内に搭載されたエクスポート機能を用いて JSON 形式で提出してもらった。録画データについては、チームごとの机の中央に 360 度カメラを設置しておき、最後のチームごとの発表を除いて、チームメンバーおよび TA とのやり取りを録画した。なお、データ収集の観点から、参加者からの質問に答える際にはその参加者の机で答えるように予め TA に伝えておいた。

## 4. データの分析

### 4.1 分析対象データの選別

3.3 節で述べた通り、ワークショップにおいて収集したデータは、Google Chrome の検索履歴、Chatbot システムのログデータ、チームメンバーや TA とのやり取りを記録した録画データの三つである。検索履歴について、チームごとの発表が終了後に提出を求めたが、参加者の一人 (P1) が大学の Google アカウントを利用しており Google Chrome の拡張機能を追加することが大学に許可されていなかったため、その一人の検索履歴のみ収集することができなかった。

Google Chrome の検索履歴について、1504 件のデータが確認できたが、そのほとんどは MakeCode のエディタや Slack 等のアプリケーションの利用によるものであったため、わからないことを調べるために検索したと見られるデータのみを選定し分析対象とした。その結果、25 件のデータが分析対象として選定された。

生成 AI の Chatbot システムのログデータについて、119 件のデータが確認できたが、その一部は以前のやり取りを参照しており、同一または関連した目的を持った入力であったため、独立した目的を持った入力のデータのみを選定し分析対象とした。その結果、80 件のデータが分析対象として選定された。

チームメンバーや TA とのやり取りを記録した録画データについて、TurboScribe<sup>\*6</sup>という AI 文字起こしサービスを利用したが、今回のデータは複数人の音声であったため、十分な精度が得られなかった。よって、主要な出来事について書き起こしたところ、291 件のデータが確認できたため、これらのデータが分析対象として選定された。

<sup>\*5</sup> <https://chromewebstore.google.com/detail/history-plus/kloodnjmhgiceceindgbfpjencnhajh>

<sup>\*6</sup> <https://turboscribe.ai/ja/>

表 1: ワークショップにおける参加者が直面した困難の内容別分類とその個数

大分類	小分類	個数	大分類	小分類	個数
SW	モジュールの使い方 (SW)	35	Idea	課題	4
	プログラムの構文エラー	22		解決策	21
	並行処理	9		アイデア→実装 (HW)	18
	コード理解	8		アイデア→実装 (SW)	37
	端子の誤参照	5	Env	コード共有	14
	コード実行時間への配慮不足	3		MakeCode の仕様	10
HW	モジュールの使い方 (HW)	11		モジュールの制約	8
	モジュールの認識	5		ワークショップによる制約	8
	端子の誤接続	3		外部環境 (PC, Wi-Fi)	8
SW+HW	欠陥箇所の誤特定	6	Other	micro:bit 自体と無関係	3
	micro:bit とパソコンの接続	4		判別不能	4

## 4.2 データの分類

3.3 節で述べた通り、本研究では「参加者が行き詰まった状態から抜け出すために他者の介入を必要とするイベント」を1つの困難と定義した [5]。そのため、まず、分析対象の各データに対し、利用した解決方法がなければ正しい知識を身につけたり正しく実装を行ったりすることができなかったと考えられるかどうかを基準に、困難または非困難のいずれであるかを分類した。

「困難」に選定されたデータに対し、その内容をもとにラベリングを行った。具体的には、データに基づいた概念抽出を目的とした木下による修正版グラウンデッド・セオリー・アプローチ [12] と呼ばれる質的分析の手法を参考にし、次のような手順で行った。1) いくつかのデータを基に分類項目を作成し、2) 作成された分類ごとにその定義や具体例を定め、3) その他のデータに対しても作成された分類項目に基づいて分類を行い、4) 分類中に例外が発生したら分類項目の再検討を行い、5) 分類項目が定着し分類が終了するまで2) から4) を繰り返した。

ここで、モジュール性の低いマイコンを対象に調査した研究 [2,3] を参考に、おおまかな分類（以下、大分類）を行ってからさらに細かな分類（以下、小分類）を行った。これらの分類項目の作成および分類作業については、複数人での議論により修正を行いながら主に筆者1名で行った。

## 5. 結果

micro:bit を用いたワークショップにおいて参加者が直面した困難を、その内容によって修正版グラウンデッド・セオリー・アプローチ [12] という手法を用いて分類した結果、表1に示す6の大分類項目および22の小分類項目が作成された。各大分類項目の定義は以下の通りである。1) SW: プログラムを記述する上での困難、2) HW: micro:bit 等の実機を繋いで組み立てる上での困難、3) SW+HW: プログラムと micro:bit との相互作用における困難、4) Idea: ア

イデアを実装に落とし込む上での困難、5) Env: 開発環境や外部環境が原因で生じる困難、6) Other: micro:bit とは関連性が低いまたは収集したデータからは判別できない困難。表1の通り、ソフトウェア (SW) およびアイデア (Idea) の困難が特に多く見られた。

## 6. 考察

本章では、これまでの分類・分析で得られたモジュール性の高いツールキットを用いた場合の困難と、先行研究により明らかになったモジュール性の低いマイコンを用いた場合の困難 [2,3,5] を比較することで、ハードウェアプロトタイピングの初学者が直面する困難に対してモジュール性が与える影響について明らかにする。

### 6.1 モジュール性が高くなっても依然直面する困難

モジュール性の低いマイコンを用いてもモジュール性の高いツールキットを用いても困難であったものとしてプログラムの構文エラーが挙げられる。これは、初学者のプログラミングの知識不足による困難であり、使用するマイコンのモジュール性の高さには関係がないことが原因だと考えられる。例えば、else if と if の違いを理解していないことによる困難や、変数の初期化を行っていないことによる困難が複数見られた。

また、端子に関する困難も依然確認できた。これには、実機とは異なる端子をプログラムで参照する場合（端子の誤参照）と接続可能ではない端子に接続する場合（端子の誤接続）の二つが含まれる。前者については、異なる端子を利用していたチームメンバーどうしでプログラムを共有した際に、使用する端子を統一し忘れることが主な原因であった。後者については、端子が規格化されてはいるものの、モジュールによって接続できる位置が決まっているにも拘らず同じ形状であるため、誤って差し込み得ることにより生じた困難であった。

## 6.2 モジュール性が高くなったことで防げる困難

モジュール性の低いマイコンを用いた場合には困難であったが、モジュール性の高いツールキットを用いた場合には困難ではなくなったのは、主にハードウェアや感情に関連するものであった。ハードウェアに関する困難について、配線やブレッドボード、回路の構成等に関する困難はモジュール性の高いツールキットを用いることで防ぐことができたが、これは、micro:bit には基本的なセンサや出力が内蔵されていることや端子が規格化されていることによると考えられる。感情に関する困難について、目に見えて興味がない、電子機器に触れない等の困難はモジュール性の高いツールキットを用いることで防ぐことができたが、これは、電子部品のモジュール性を高めることで自己効力感が向上するという Sadler らの調査結果 [5] とも合致している。

## 6.3 モジュール性が高くなったことで新たに生じる困難

モジュール性の低いマイコンを用いた場合には困難ではなかったが、モジュール性の高いツールキットを用いることで新たに生じてしまった困難の多くは、ソフトウェアおよびアイデアに関するものであった。ソフトウェアに関する困難については、「モジュールの使い方 (SW)」や「並行処理」等が挙げられる。「モジュールの使い方 (SW)」とは、センサや出力の機能を利用するためのコードを知らないことによる困難を指し、micro:bit のスピーカーの音量を調整できるかわからない場合や、Grove Kit の LED テープで一度点灯した LED を消すことができない場合が例として挙げられる。これらは、センサや出力の機能ごとに関数が定義されるようになったことが原因であると考えられるが、容易に解決できる場合も多い困難であった。

一方で、「並行処理」とは、複数の処理を（見かけ上）同時に行おうとして生じた困難を指すが、これは容易に解決できない場合が多い困難であった。「並行処理」の困難が新たに生じてしまった理由は、並行処理の知識なしで複数の処理を同時に行うようなコードを記述することができてしまうことにあると考えられる。一般的に、モジュール性の低いマイコンでは上から 1 行ずつ順にコードが実行されるのに対し、モジュール性の高いツールキットでは複数の処理を見かけ上同時に行うことが可能である。例えば、「ずっと」というループのコードブロックと「ボタン A が押されたとき」という入力のコードブロックを同時に扱うことができ、常に繰り返しコードを実行する間もボタンを押すという入力を受け付けることができる。だが、一部のコードブロックは実行に時間がかかるため後の処理を阻害してしまうことがある。例えば、明るさセンサで読み取った値を常に micro:bit の LED 画面に表示する場合、右から左へ流れるように数値が順に表示されるが、この間は明るさセン

サの新たな値の取得が行われない。

「並行処理」の困難の例として、音楽のバックグラウンド再生に関するものも複数見られた。micro:bit のスピーカーで音を鳴らす際には「終わるまで」、「バックグラウンドで一度だけ」、「バックグラウンドですっと」という 3 つのモードがあり、音を鳴らし終えてから次のコードブロック以降の処理を行うか、音を鳴らし始めると同時に次のコードブロック以降の処理を行うかを選ぶことができる。このことはワークショップの講義においても、例を示して説明した。だが、バックグラウンドで音を鳴らすコードを繰り返し実行するようなコードを記述してしまうことで、想定外のメロディーが流れてしまうという困難に直面した初学者もいた。

一方で、アイデア (Idea) に関する困難については、表 1 の通り、「課題を考える部分」よりも、その後の micro:bit を用いた解決策を考え、実際に作るためにどのようなモジュールやコードを利用すればよいのかを考えるという「アイデアを実装に落とし込む部分」で主に困難に直面するという結果が得られた。これは、モジュール性の低いマイコンを用いる場合には、主に初学者は LED の点滅等のタスクを行うだけであったため [3,5]、モジュール性による単純な比較は行えないが、6.2 節の通り、モジュール性が高まったことで初学者が直面する多くの困難を防ぐことができるため、自分でデバイスを制作するうえで初学者が多種多様なアイデアを考え得る余地が生まれたことが要因であると考えられる。

## 6.4 ハードウェアプロトタイピングにおける初学者の支援

これまでモジュール性の高いツールキットを用いた困難の特性およびモジュール性による困難の違いについて調査・分析してきたが、それらの結果を踏まえてモジュール性の高いツールキットを用いたハードウェアプロトタイピングにおいて初学者をどのように支援できるのかについて検討する。

### 6.4.1 初学者が困難に直面する際の解決方法

本研究のワークショップでは、参加者が困難に直面した際に 4 つの解決方法を利用することができるようにした。その結果、参加者は困難に直面した際に Google Chrome で検索する傾向がかなり低く、アイデアを実装に落とし込むまでに何かかわからないことがあった場合には生成 AI を利用する傾向が高く、また、実際に組んだプログラムを実行して想定外の挙動を示した場合にはチームメンバーや TA に質問する傾向が高いことがわかった。よって、このように電子工作の初学者がハードウェアプロトタイピングを行う際には、アイデアを実装に落とし込む部分においては生成 AI の利用を促し、デバイスを実装する際には複数人でのチームで行うように設計することが望ましいと考えられ



る [8].

だが、チームでの共同制作では、互いに頭の中で考えていることの意味疎通を図ることが初学者どうしでは難しいという意見もあり、単に生成 AI の利用を促すだけではなく、チーム内で話の仲介役を担当し互いの考えを言語化してまとめるために、チームごとに TA や生成 AI の Chatbot をファシリテータとして設けることが解決策として考えられる。

複数人でのプロダクト開発に際して、電子工作の初学者同士では特に、各人が何となく頭に浮かんでいるコードの設計や作業段階の分け方、気にすべきだと思うポイントなどを適切に伝え合うことが難しいと感じました。(P3)

#### 6.4.2 開発環境に起因する困難を防ぐために求められる条件

6.4.1 項でも述べた通り、電子工作の初学者にとって、困難に直面した際にチームメンバーを頼ることができることは有用であると考えられる。だが、今回のワークショップでは、表 1 の通り、開発環境 (Env) に関する困難も数多く見られた。特に、コードの共有に関する困難が多く、制作したプログラムをチームメンバーどうしで統合したいが、異なるタブにコードブロックをコピーアンドペーストできないという内容であった。

また、そもそも、micro:bit の代表的なエディタである MakeCode のファイルの共有は非同期であり、共有されたプログラムを閲覧・編集することはできるが、一方がプログラムに変更を加えても、もう一方にその変更内容が反映されるわけではない。チームメンバーどうしで協力してデバイスを実装するような場面では、リアルタイムで共同編集することができるような micro:bit のエディタが必要とされているのではないかと考えられる。これによって、チームメンバーどうしでのプログラムの統合そのものが困難ではなくなるのみならず、6.1 節で述べたようなプログラム統合時の端子や変数の統一を忘れるという困難も防ぐことができる。

さらに、MakeCode でブロックベースのプログラミングを行う場合、生成 AI に質問する際に一度 JavaScript を介する必要がある、ブロックで説明してほしいという内容のプロンプトを入力した参加者も見られた。ブロックベースのコードを参照しながら生成 AI に質問できるような開発環境が求められていると考えられる。

五分前と起床時間には鳴らすけど、それ以降、3 回目、4 回目のアラームが鳴るかはランダム化するには？ ブロックで説明して (P9)

#### 6.4.3 モジュール性の高さに起因する困難を解決するシステム

6.3 節で述べた通り、モジュール性の高いツールキットを用いることで、初学者は並行処理の知識を特に必要とすることなく複数の処理を同時に行うようなコードを簡単に記述することができるが、想定外の挙動を示した場合に何が原因なのか、仕組みが全く分からないため対処することができない。また、各コードの実行にはわずかな時間を要するということを考慮してプログラムを書いていないことで、想定外の挙動を示し解決に時間を要する例も複数見られた。そのため、解決策の一つとして、ハードウェアプロトタイピングの初学者に対する講義のなかで「並行処理」の注意点を事例とともに扱うことが考えられる。また、どのブロックが原因で他の処理が行われなくなっているのかわかるように、実行しているコードがリアルタイムで可視化されるような micro:bit 用のシステムというのも解決策として考えられる。

### 7. 制約と展望

#### 7.1 データの分類の客観的妥当性

本研究におけるデータの分類作業は主に 1 名のみで行った。分類の妥当性を確保するために、修正版グラウンデッド・セオリー・アプローチ [12] を用いて分類項目の作成を行い、一部のデータについては複数人での議論を踏まえて分類を行ったが、依然として分類の客観的妥当性に関しては十分に保証されているとは言い難い。さらなる客観的妥当性を保証するためには、複数人で同じ分類項目の定義に従って分類を行い、その結果を比較・検討する必要がある。

#### 7.2 ワークショップの設計方針

##### 7.2.1 ワークショップの参加者について

本研究のワークショップでは、参加者として電子工作の経験がない成人を対象に募集し、応募時に電子工作の経験について回答してもらい、その内容をもとに適切だと判断した 15 人 (男:9 人, 女:6 人) の希望者に参加してもらった。その内訳としては、学生が 13 人 (大学数は 9)、非学生が 2 人 (P11, P15) であった。学生の学部については、文系が 7 人、理系が 6 人であり、学年については、学部 1 年が 2 人、学部 2 年が 2 人、学部 3 年が 3 人、学部 4 年が 3 人、修士 1 年 (または学部 5 年) が 3 人であった。一方で、電子工作の経験が全くないわけではない参加者が 3 人いた。その 3 人の電子工作経験と参加を許可した理由は以下の通りである。

**P2** Arduino (3 ヶ月)、変数や繰り返しを理解していない文系学生のため。

**P4** micro:bit (授業で使用)、プログラミングの授業経験 1 つで、文系学生のため。

**P13** Raspberry Pi (1 ヶ月), プログラミングの授業経験  
1 つで, micro:bit 経験がないため.

ワークショップにおける上記の 3 人の言動や振る舞い, 直面する困難の数等が他の 12 人と著しく異なるわけではないことからデータの一般性を欠くことはないと判断し, 電子経験がほとんどない初学者とみなしてデータに含めた. だが, 電子工作の初学者を対象とした調査において, 多少なりとも電子工作の経験がある者を対象としたことは本研究の結果の制約となり得る. また, サンプル数の少なさおよび成人に限定していることも制約となり得る. 理想的には, 電子工作の経験が全くない幅広い年齢層を対象として多くの人数のデータを収集することが求められる.

### 7.2.2 ワークショップの構成について

本研究のワークショップでは, 制作するデバイスについて参加者にアイデアを出してもらいそのアイデアの実装を行い発表するところまで一通りのものづくりの流れを行ってもらった. だが, チームによって作品を制作する難易度や必要な知識等がアイデアに依存することに留意すると, 参加者の少なさゆえに, 初学者が直面する困難の一般性が保証されているとは言い難い. より一般的な困難を調査するためには, 多くの人数のデータを収集する必要がある.

### 7.2.3 ワークショップで利用した生成 AI の Chatbot について

本研究のワークショップでは, 参加者が困難に直面した際の解決方法の一つとして, 研究者側で生成 AI の Chatbot を用意した. だが, ファイルの入出力等はいえず, テキストベースでのやり取りのみの Chatbot であった. 6.4.1 項で述べたように, デバイスを実装する際には, 直面した困難を解決する方法として生成 AI を利用するよりもチームメンバーや TA に質問することの方が多かったが, これは, Chatbot が画像や音声の入出力に対応しておらず, 質問をテキストベースで入力するのに手間と時間が掛かることが原因である可能性がある.

また, 6.4.2 項で述べたように, 生成 AI の Chatbot にブロックベースのコードでの説明を求めた参加者も複数見られた. つまり, 電子工作の初学者にマルチモーダルな生成 AI の Chatbot を提供した場合, アイデアを実装に落とし込む困難だけではなく, 実装する際に直面するソフトウェア・ハードウェアの両方に関する困難について, チームメンバーや TA に質問するのと同じように生成 AI を利用する可能性があり, ハードウェアプロトタイピングの初学者をどのように支援すべきなのかを明らかにするためにも調査する価値が大いにあると思われる.

### 参考文献

- [1] Baldwin, C. and Clark, K.: *Design Rules, Volume 1: The Power of Modularity*, MIT Press (2000).
- [2] Booth, T., Stumpf, S., Bird, J. and Jones, S.: Crossed

- Wires: Investigating the Problems of End-User Developers in a Physical Computing Task, *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, New York, NY, USA, Association for Computing Machinery, p. 3485–3497 (online), DOI: 10.1145/2858036.2858533 (2016).
- [3] DesPortes, K. and DiSalvo, B.: Trials and Tribulations of Novices Working with the Arduino, *Proceedings of the 2019 ACM Conference on International Computing Education Research*, ICER '19, New York, NY, USA, Association for Computing Machinery, p. 219–227 (online), DOI: 10.1145/3291279.3339427 (2019).
- [4] Horn, M. S., Crouser, R. J. and Bers, M. U.: Tangible interaction and learning: the case for a hybrid approach, *Personal and Ubiquitous Computing*, Vol. 16, No. 4, pp. 379–389 (online), DOI: 10.1007/s00779-011-0404-2 (2012).
- [5] Joel Sadler, L. S. and Blikstein, P.: Building blocks in creative computing: modularity increases the probability of prototyping novel ideas, *International Journal of Design Creativity and Innovation*, Vol. 5, No. 3-4, pp. 168–184 (online), DOI: 10.1080/21650349.2015.1136796 (2017).
- [6] Kafai, Y. B., Lee, E., Searle, K., Fields, D., Kaplan, E. and Lui, D.: A Crafts-Oriented Approach to Computing in High School: Introducing Computational Concepts, Practices, and Perspectives with Electronic Textiles, *ACM Trans. Comput. Educ.*, Vol. 14, No. 1 (online), DOI: 10.1145/2576874 (2014).
- [7] Katterfeldt, E.-S., Dittert, N. and Schelhowe, H.: Designing digital fabrication learning environments for Bildung: Implications from ten years of physical computing workshops, *International Journal of Child-Computer Interaction*, Vol. 5, pp. 3–10 (online), DOI: <https://doi.org/10.1016/j.ijcci.2015.08.001> (2015). Digital Fabrication in Education.
- [8] Marshall, P.: Do tangible interfaces enhance learning?, *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*, TEI '07, New York, NY, USA, Association for Computing Machinery, p. 163–170 (online), DOI: 10.1145/1226969.1227004 (2007).
- [9] Schneider, B., Jermann, P., Zufferey, G. and Dillenbourg, P.: Benefits of a Tangible Interface for Collaborative Learning and Interaction, *IEEE Transactions on Learning Technologies*, Vol. 4, No. 3, pp. 222–232 (online), DOI: 10.1109/TLT.2010.36 (2011).
- [10] Sentance, S., Waite, J., Hodges, S., MacLeod, E. and Yeomans, L.: "Creating Cool Stuff": Pupils' Experience of the BBC micro:bit, *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '17, New York, NY, USA, Association for Computing Machinery, p. 531–536 (online), DOI: 10.1145/3017680.3017749 (2017).
- [11] Sentance, S., Waite, J., Yeomans, L. and MacLeod, E.: Teaching with physical computing devices: the BBC micro:bit initiative, *Proceedings of the 12th Workshop on Primary and Secondary Computing Education*, WiPSCE '17, New York, NY, USA, Association for Computing Machinery, p. 87–96 (online), DOI: 10.1145/3137065.3137083 (2017).
- [12] 木下康仁: 修正版グラウンデッド・セオリー・アプローチ (M-GTA) の分析技法, 富山大学看護学会誌, Vol. 6, No. 2, pp. 1–10 (オンライン), DOI: 10.15099/00002599 (2007).