2021 年度 九州工業大学 情報工学部 卒業論文 概要

所 属 学 科	知的システム工学科	指 導 教 員	田中和明　准教授
学 生 番 号	20223012	氏　　名	村上 旭人
論 文 題 目	小型デバイス向けのデータフロー型 プログラミング環境の構築		

1　緒言

教育機関ではプログラミング科目が必修化されている．プログラミング科目の目的は，プログラミングロジックの理解が主である．そのため，教材としてブロック型のビジュアルプログラミング言語 (VPL) がよく用いられている．近年，内閣府より科学技術政策 Society5.0 が定められ，IoT や AI といったデータの活用が必要技術要素として挙げられている．特に IoT ではデータの流れ (データフロー) の理解が重要である．しかし，主流となっているブロック型 VPL ではデータフローの理解は難しいため，プログラミングロジックの理解と，データの活用の理解は異なる観点での教育方法が必要であると考える．

そこで本研究では，マイコンボードを利用した IoT 開発に対し，デバイス間の情報の流れを直感的に理解できるプログラミング環境を構築することを目的とする．ユーザーが視覚的な操作によりプログラムを作成した後，マイコンボード用のプログラムを自動生成する．これにより，プログラミング初心者でも IoT 開発が行いやすくなり，開発効率の向上にも繋がる．

2　システム構成

2.1　Node-RED

Node-RED[1] は，IBM により開発されたデータフロー型ビジュアルプログラミング開発ツールである．プログラミング方法は，ノードと呼ばれる各機能がまとめられたブロックを配置していき，これらを線で繋ぎ合わせてフローを作成し，ノード同士でデータを送信・受信することで様々な処理を行わせることができる．また，HTML と Javascript を用いてノードの自作も可能である．本研究ではこれらの特徴を持つ Node-RED を活用していき，プログラミング環境構築を目指す．

2.2　Ruby・mruby/c

mruby/c[2] は Ruby の特徴を引き継いだ組込み開発向け言語である．Ruby コードを mruby コンパイラによりコンパイルしてバイトコードに変換し，VM (バーチャルマシン) で実行することでメモリ消費量を削減している．これにより，組み込み分野でも Ruby コードによる記述で開発が行えるようになった．本研究では Ruby と mruby/c を用いている．理由は，組み込み開発で使われている C 言語と比較すると，可読性と生産性に優れており，プログラミング初学者に理解されやすいプログラム言語であるため教育用途で非常に適していると考えた．

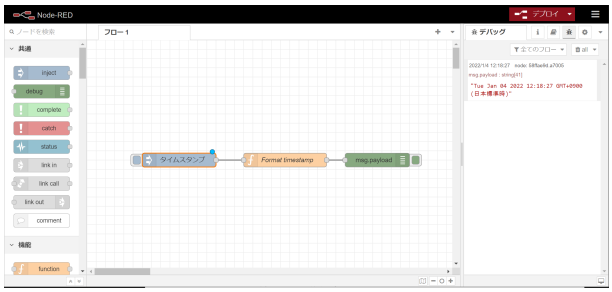

図 1: Node-RED エディタ画面

3　研究内容

3.1　システム概要

本研究で開発するシステム構成を図 2 に示す．手順としては，まずユーザーは Node-RED 上でマイコンボード用ノードを使ってフローを作成していく．作成したフローは JSON ファイルで保存されているため，その JSON ファイルを抽出する．その後，Ruby コード生成器プログラムを実行させ，抽出した JSON ファイルを基に，マイコンボードが動作する Ruby コードを自動生成する．最後に mruby コンパイラにより mruby/c バイトコードに変換させ，マイコンボードに実装する．

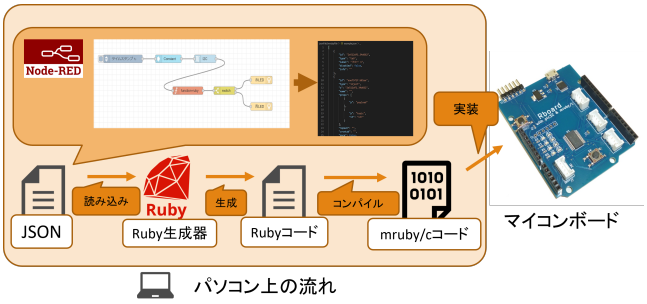

図 2: フロー作成からマイコンへの実装の流れ

3.2　マイコンボード用ノードの開発

マイコンボードの制御に関して必要なノードを開発した．開発したノードと一部のノードの編集ダイアログを図 3 に記載する．ノードの外観や編集ダイアログのデザインや設定項目の情報管理などを HTML と

Javascript を用いて作成した．

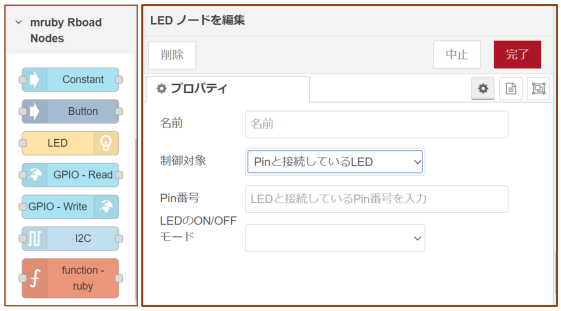

図 3: (左) 自作ノード一覧 (右)LED ノードの編集ダイアログ

3.3 マイコンボードへ実装する Ruby コードの生成

Node-RED で抽出した JSON ファイルには，配置されたノードの種類，ノード同士の接続関係，ノードの識別 ID や設定情報といったノードのプロパティ情報が記録されている．Ruby コード生成器はこの JSON ファイルに基づき，Ruby コードの生成を行う．Ruby コード生成器の処理手順を図 4 のフローチャートに示す．

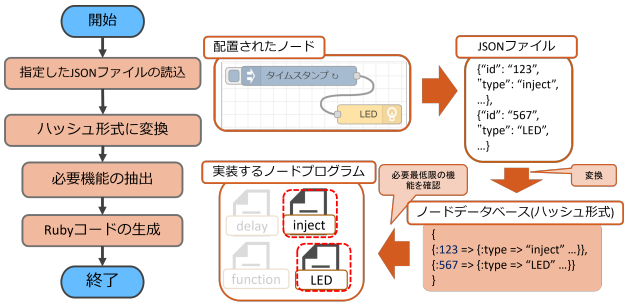

図 4: Ruby コード生成手順

まず，JSON ファイルを読み込み，Ruby で扱えるように図 4 右下図のようにハッシュ形式のデータに変換する．以降はこのノードデータベースを用いて，Ruby コードを生成していくことになる．必要機能の抽出作業では，ノードデータベースからノードのタイプを抽出し，抽出されたタイプに応じてノードプログラムを選択する．ノードプログラムとは，ノードのタイプごとに機能をまとめたプログラムであり，事前に用意されている．最後に，ノードデータベースやノード間のデータ制御機能といった基本機能と，選択されたノードプログラムをまとめて Ruby コードとして生成する．生成される Ruby コードの構成を図 5 に示す．この構成により，Node-RED 上でユーザーが想定した動作を実現することができる．

4 動作検証

4.1 温湿度センサによる LED 制御のプログラム作成

温湿度センサ (SHT31) を使い，温度による LED 制御を行うプログラムを作成し，RBoard[3] による動作

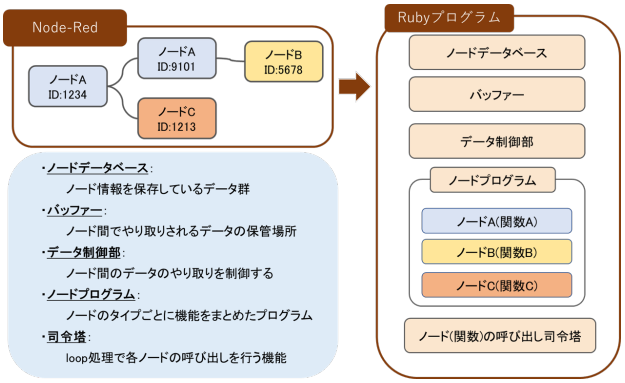

図 5: 生成される Ruby コードの構成

検証を行った．Node-RED で作成したプログラムを図 6 に示す．このプログラムは摂氏温度が 24 度以上であれば赤色 LED が点灯し，24 度未満であれば青色 LED が点灯する．`inject` ノードは `Constant` ノードを通して 1 秒ごとにデータを送信する．受信した `I2C` ノードは温湿度センサにバイトコードのコマンドを送信し，センサから温度値を受け取る．しかし，受け取った温度値はバイトコードであるため摂氏温度への変換処理を `function-ruby` ノードによる自由記述で実装する．その後，温度値を `switch` ノードに送信し判定を行い，LED 制御を行う．

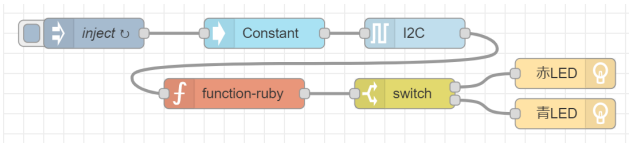

図 6: 作成したフロー 温度センサによる LED 制御

4.2 動作結果

図 6 のフローを基に生成された Ruby コードをコンパイルし，バイトコードを RBoard に送信した結果，想定通りの動作を行った．

5 結言

4.2 項より，データフロー型ビジュアルプログラミング言語を用いたプログラミング環境の構築に成功した．これにより，ユーザーはデータフローを意識しながらプログラム開発が行えるようになり，プログラミング初学者の IoT 開発が容易になったと考えられる．

今後は，RBoard だけでなく他マイコンボードでも動作できるように Ruby コード生成ルーチンの切り替え機能の実装を検討していく．

参考文献

[1] Node-RED ホームページ URL:https://nodered.jp/
[2] mruby/c しまねソフト研究開発センター URL:https://www.s-itoc.jp/activity/research/mrubyc/
[3] RBoard 島根情報処理センター URL:https://www.sjc-inc.co.jp/service/rboard