C-025

JuliaMBD の機能拡張と実用性検証に関する研究
Research on functionality enhancement and practicality verification of JuliaMBD

中山 昂紀[1)]　岡村 寛之[1)]　土肥 正[1)]
Koki Nakayama　Hiroyuki Okamura　Tadashi Dohi

1　はじめに

組み込みシステムとは，主に物理デバイスを制御するソフトウェアシステムであり，自動車製品や家電製品をはじめとして様々な製品で利用されている．一般的に，組み込みシステムは我々の生活に物理的に影響を与えるため，より高い信頼性や安全性を確保することが必要となる．一方で，組み込みシステムは開発環境と実行環境が異なるため，構築したシステムをテストする際には，ターゲットとするマイコンなどを有した実行環境をハードウェアとして準備し，その上でテストを実行しなければならない．またエラーの発見についても実行環境における電気的な信号を確認するなど，通常のソフトウェア開発にない難しさが存在する．

近年，組み込みシステム開発に対してモデルベース開発（Model-Based Development; MBD）が注目されている [1]．MBD では，組み込みシステムが稼働する実行環境をモデル化し，ハードウェア的な実行環境を準備することなく，構築したシステムの動作をモデル上でソフトウェア的にシミュレーションすることにより，テストを実行する．この手法は，開発の初期段階で機能要求と非機能要件が満たされるかどうかが確認でき，ハードウェア的な実行環境におけるテスト工数を低減させることができ，結果として高い品質の組み込みシステムを迅速に開発することが可能となる．

しかしながら，MBD にはいくつかの問題点が存在する．その一つとしてベンダーロックインの問題がある．ベンダーロックインとは，システムの開発が特定のベンダーが提供する技術に依存し，他のベンダーに移行することが難しくなる状況を示している．MBD におけるシミュレーション環境は，Mathworks 社が提供する MATLAB/Simulink がデファクトスタンダートとなっている．特に Simulink はブロック図を用いたモデルの構築とシミュレーションが容易に実行できることと，作成したモデルからプログラムを自動生成する機能なども有していることから，自動車における組み込みシステム開発に広く利用されている．その一方で，MATLAB/Simulink は Mathworks 社によるライセンス管理が行われているため保守料金が発生する．そのため，他のベンダーに移行しようとしてもモデル互換性などから実質的な移行が難しいといった問題がある．このような背景から，MBD のより広範な発展のためには，オープンなライセンスに基づいた MATLAB/Simulink 相当の環境が必要とされている．

文献 [2] では，Julia 言語により，Simulink と同等な機能を有するパッケージ（JuliaMBD）の開発を行っている．Julia とは MIT ライセンスに基づくオープンな言語ならびに言語実行環境であり，数値計算分野で発展が見込まれるプログラミング言語である．JuliaMBD では，Julia 言語における強力なマクロ機能を利用して，Simulink のようなブロック構成から，システムをシミュレーションするための式の構築を行うことができる．構築された式は Julia が提供する微分方程式ソルバーを用いて容易にシミュレーション実行することができる．本稿では文献 [2] で開発された JuliaMBD の拡張ならびに実用性の検証を行う．具体的には，JuliaMBD における GUI 機能の強化を行い，JAMBE (Japan Automotive Model-Based Development Center) が提供するモデルを用いた有効性の検証を行う．

```
@model Test begin
  @parameter begin
      R
      L
      C
      voltage
  end
  @block begin
      system = RLC(R=R, L=L, C=C)
      source = Step(steptime=0.1,
                    finalvalue=voltage)
  end
  @connect source.out => system.in
  @scope begin
      source.out => v
      system.out => i
  end
end
m = @compile Test(R=10, L=100e-3,
                  C=10e-6, voltage=5)
result = simulate(m, tspan=(0.0, 60.0))
plot(result)
```

図 1　DSL の例．

2　JuliaMBD

JuliaMBD [1)] は Julia パッケージとして，(i) ブロックによるシステム構築，(ii) 構築したシステムのシミュレーション実行の機能を提供する．ブロックによるシステム構築では，Julia のマクロ機能を使ってシステムを構築するための DSL (Domain Specific Language) を定義している．DSL によるシステム記述を元にして，システムをシミュレーションするための式を定義し，Julia パッケージとして提供される DifferentialEquation を用いて定義した式から微分方程式を解くことでシミュレーション実行することができる．図 1 は JuliaMBD における DSL の例である．`@model` マクロでブロックや接続の定義を行い，`@compile` マクロで式の構築を行う．`simulate` 関数で微分方程式を解くことでシミュレーションを実行し，`plot` 関数で結果のグラフを表示する．

本研究における主な機能拡張として，上記のような DSL を図から生成する機能を作成した．具体的には Diagrams.net と呼ばれるドローツールで作成したブロック図から上記の DSL を生成する機能を作成した．図 2 は Diagrams.net で作成したブロック線図の例である．あらかじめ定義された Diagrams.net のコンポーネントを提

1）　広島大学院先進理工系科学研究科

1）　https://github.com/JuliaMBD/JuliaMBD.jl

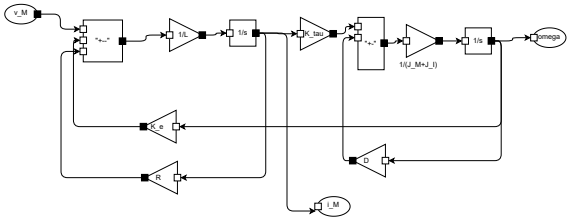

図 2　ブロック図の例．

表 1　コンパイル時間とシミュレーション実行時間の比較（単位：秒）．

	JuliaMBD	Simulink
コンパイル時間	3.0	14.35
実行時間	13.42	4.05

供しており，DSL よりも容易にモデルを構築することができる．

3　実用性検証

本稿では，JuliaMBD に対する実用性検証を行う．検証には JAMBE が提供している「振動モデル」を用いた．これは，路面からの値から車両の振動をシミュレーションする Simulink によるモデルである．全体で 462 個のブロックから構成されている．本検証を行うにあたり，これまで実装されていなかったいくつかのブロックを JuliaMBD 上に実装した．また，モデルは Diagrams.net を用いてブロック定義を行い，それらを DSL に変換することでシミュレーション実行を行った．

表 1 は JuliaMBD におけるコンパイル時間（DSL から式を構築するまでの時間）と実行時間（微分方程式を解き，グラフを表示するまでの時間）と，Simulink によるコンパイル時間とシミュレーション実行時間を示している．この表から，JuliaMBD は，Simulink より実行時間がかかってしまうものの，コンパイル時間を含めるとほぼ同等の実行時間でシミュレーションできていることがわかる．また，Simulink の方がコンパイル時間が長くなっているのは，Julia よりも多くの最適化を行っているための考えられる．その結果として，シミュレーション時間については JuliaMBD よりもかなり高速に実行できている．一方，JuliaMBD は Julia の仕組み上，実行時にコンパイルする JIT (Just-In-Time) コンパイルと呼ばれる仕組みを採用しているため，実行時間が長くなる傾向がある．

次に，出力結果についての検証を行う．図 3 および図 4 はそれぞれ Simulink および JuliaMBD によるシミュレーション結果を示している．結果については概ね同じ結果を示しているが，一部異なる結果を示している．特に，初期時刻周辺においては，JuliaMBD の方が大きく結果が振動している．これは，現状の JuliaMBD の実装において微分（Derivative）ブロックが不十分であるためであり，モデル内に高階微分（二階微分以上）が含まれている場合に異なる結果となる．今回検証した実践的なモデルにおいても高階微分が使われていることから，今後対応する必要がある．

4　まとめ

本稿では，JuliaMBD に関する紹介を行い，GUI 機能の強化ならびに実践的なモデルを用いた検証を行った．

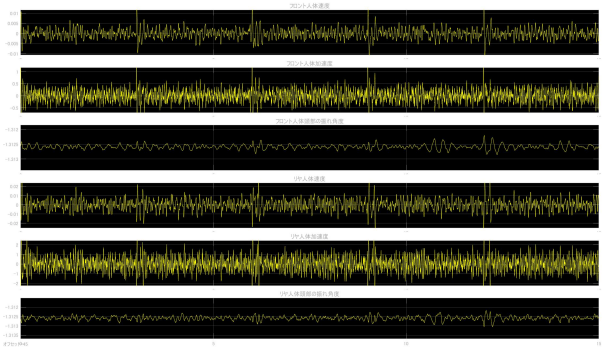

図 3　Simulink による出力結果．

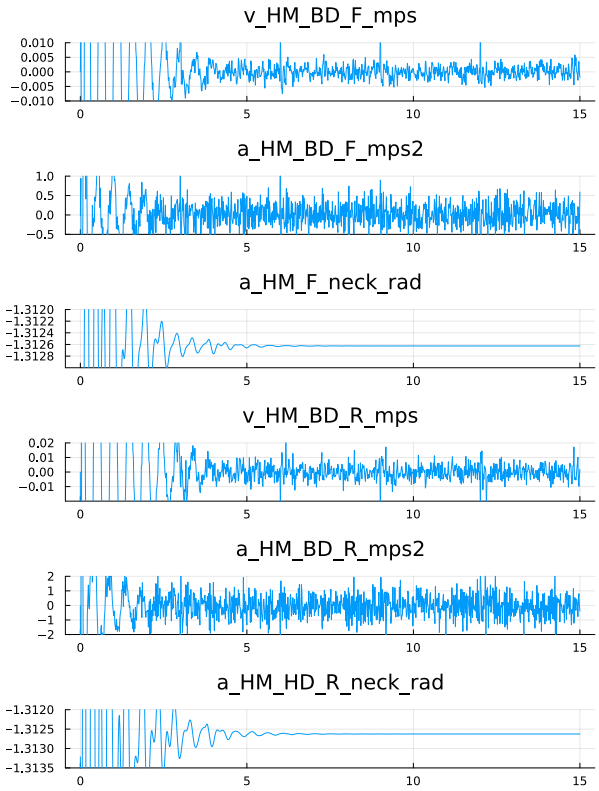

図 4　JuliaMBD による出力結果．

モデル構築やコンパイルに関しては，Simulink と比較して大きく劣っている点はなかった．一方で，シミュレーション結果については一部のモデル記述に対応できていない点があるため，今後の修正が必要となる．また，モデルからマイコン実行のためのバイナリを生成する機能についても検討を行う予定である．

参考文献

[1] 山本透, 脇谷伸, 原田靖裕, 香川直己, 足立智彦, 沖俊任, 原田真悟, 改訂 実習で学ぶモデルベース開発 -「モデル」を共通言語とする V 字開発プロセス -, コロナ社，2023.

[2] 永田裕人, 岡村宏之, 土肥正，Julia 言語を用いた MATLAB/Simulink クローンの試作, 第 71 回電気・情報関連学会中国支部連合大会講演論文集, ROMBUNNO.R20-25-01-02, 2020.