

Embedded Coderによる コード自動生成 & ソフト開発の効率化

MathWorks Japan

アプリケーションエンジニアリング部

はじめに

本セッションの対象者

- バッテリーマネジメントシステムの開発に興味がある方
- 本システム開発時のソフトウェア検証・実装方法に興味がある方

本セッションでお伝えしたいこと

- 自動コード生成によるソフトウェア開発
- バッテリーマネジメントシステムのソフトウェア検証・実装方法

アジェンダ

- 背景
- マイコン向け量産コード生成の紹介
 - コード生成ツール Embedded Coderの概要
 - バッテリーマネジメントシステム開発への適用例
- シミュレータによるコントローラの機能評価の紹介 (HILS)
 - MathWorksのRCP/HILSソリューションの概要
 - バッテリーマネジメントシステム開発への適用例

アジェンダ

- 背景
- マイコン向け量産コード生成の紹介
 - コード生成ツール Embedded Coderの概要
 - バッテリーマネジメントシステム開発への適用例
- シミュレータによるコントローラの機能評価の紹介 (HILS)
 - MathWorksのRCP/HILSソリューションの概要
 - バッテリーマネジメントシステム開発への適用例

背景

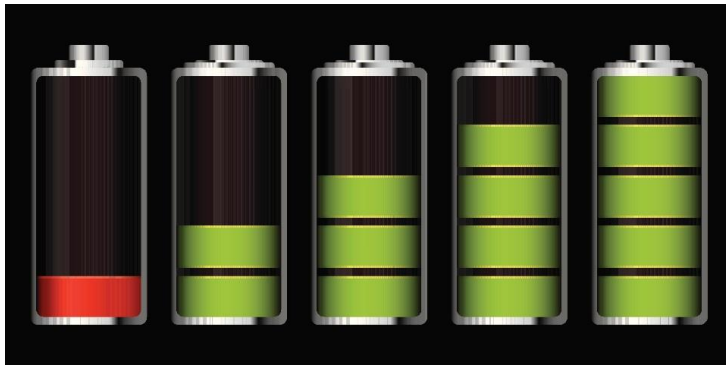
- 近年、大容量のバッテリーが様々な所で使われている。
- 特に、航空機や車両など、一つの不具合で大事故に発展しかねない機器にも搭載されている。



バッテリーマネジメントシステム開発のポイント

- 複雑なシステム特性に対処するソフトウェア開発が必要

非線形特性



経年劣化



温度依存性



バッテリーシステム開発へのモデルベースデザインの適用

バッテリーマネジメントシステム開発の課題：

- 複雑なシステムに対して多数のソフトウェア機能構築が必要
- 構築したソフトウェア機能の効率的な実装手法が必要



モデルベースデザイン適用時に見込める効果：

- シミュレーション技術によるソフトウェア機能の早期検証
- 自動コード生成技術によるソフトウェア機能の効率的な検証・実装

バッテリーシステム開発へのモデルベースデザインの適用

バッテリーマネジメントシステム開発の課題：

- 複雑なシステムに対して多数のソフトウェア機能構築が必要
- 構築したソフトウェア機能の効率的な実装手法が必要



モデルベースデザイン適用時に見込める効果：

- シミュレーション技術によるソフトウェア機能の早期検証
- 自動コード生成技術によるソフトウェア機能の効率的な検証・実装

実装デバイス・用途に向けた自動コード生成関連製品

MCU/ECU



C / CPP



M-ファイルの量産コード生成

- MATLAB Coder
- Embedded Coder

固定小数点演算 & コード生成

- Fixed-Point Designer



Simulinkモデルの量産コード生成

- MATLAB Coder
- Simulink Coder
- Embedded Coder

AUTOSARモデリング & コード生成

- AUTOSAR Blockset(R2019a~)
- R2018b以前はAUTOSAR用Embedded Coder Support Packageを提供

本日紹介する内容

RCP/HILS



- MATLAB Coder
- Simulink Coder
- HDL Coder

C/HDL

例) Speedgoat, dSPACE, A&D, 等

FPGA/SoC



- HDL Coder
- Fixed-Point Designer
- MATLAB Coder

HDL

例) Xilinx Zynq

GPU



- GPU Coder
- MATLAB Coder
- Simulink Coder

CUDA

例) Nvidia Jetson

※基本製品であるMATLAB/Simulink/Stateflowは除く

アジェンダ

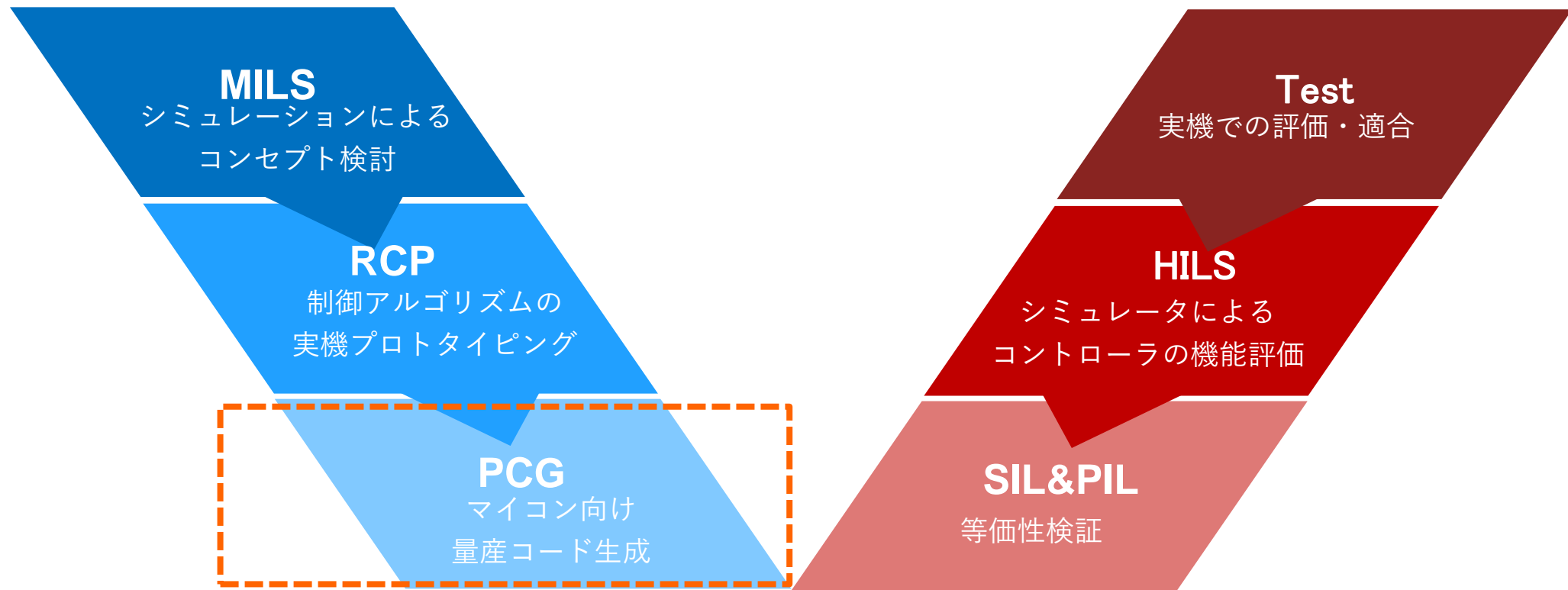
- 背景

- マイコン向け量産コード生成の紹介
 - コード生成ツール Embedded Coderの概要
 - バッテリーマネジメントシステム開発への適用例

- シミュレータによるコントローラの機能評価の紹介 (HILS)
 - MathWorksのRCP/HILSソリューションの概要
 - バッテリーマネジメントシステム開発への適用例

モデルベースデザインにおけるVプロセス

- コンセプト検討から実装・実機評価まで利用可能なソリューションが存在



MILS: Model In the Loop Simulation

RCP: Rapid Control Prototyping

PCG: Product Code Generation

SILS: Software In the Loop Simulation

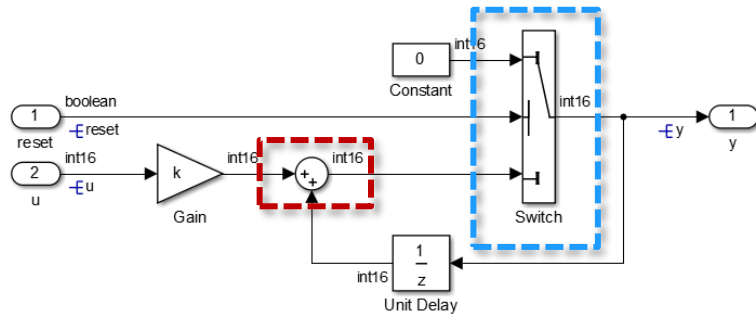
PILS: Processor In the Loop Simulation

HILS: Hardware In the Loop Simulation

Embedded Coder :

モデル/M-ファイルから量産/組込み用C/C++コードを自動生成します

モデル (ロジック)



コード
自動生成

```
boolean_T reset;  
const volatile int8_T k = 2;
```

```
void rst_cntr_step(void)
```

```
{  
    if (reset) {  
        y = 0;  
    } else {  
        y += (int16_T)(k * u);  
    }  
}
```

C or CPP

ソフト仕様



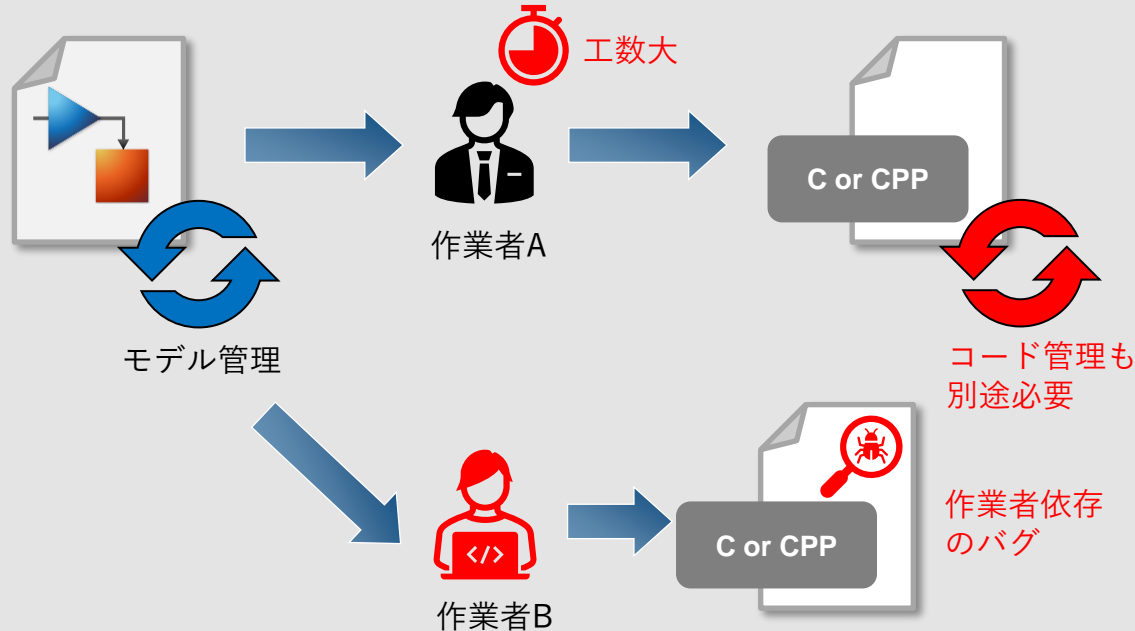
- 関数
 - 名前、引数
- 変数/定数属性
 - 名前、記憶クラス、データ型
- コード書式

モデルベースデザインでの自動コード生成導入のメリット

- 自動コード生成はコード開発のQCD向上に寄与します

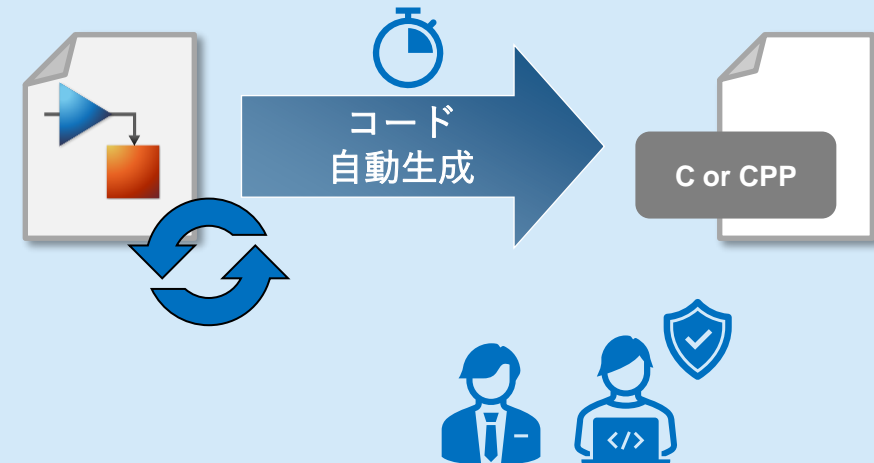
自動コード生成導入前

- 【納期】 モデルを理解したコード作成が大変...
- 【コスト】 モデル/コードの二重管理...乖離リスク...
- 【品質】 作業員依存のコード品質...バグ混入リスク...



自動コード生成導入後

- 【納期】 コード作成工数を大幅に削減できる
- 【コスト】 モデルの一元管理により整合性を保証
- 【品質】 作業員に依らないコード品質を実現



モデル自動生成コードの性能

- ハンドコードと遜色ないCコードを生成可能
 - 一般論として、最適化されたハンドコードと比べてメモリ効率性・計算速度は劣る可能性があります
 - 高効率コード生成にはツール知識やモデリング上の工夫が必要

Visteon社パワートレイン制御ソフトの例：

		Code Size
Hand Code		928
Auto Code	No overflow/underflow check	904
	Check OF/UF everywhere	1562
	Check only where necessary	934

*Based on Tasking Compiler for ST10

Table 2 ROM and RAM comparison between a floating-point hand code and auto code.

	Hand Code	Auto Code
ROM	6408	6192
RAM	132	112

※SAE Technical Paper 2004-01-0269, March 2004

メモリ効率性が同等

Delphi社HVモータ制御ソフトの例：

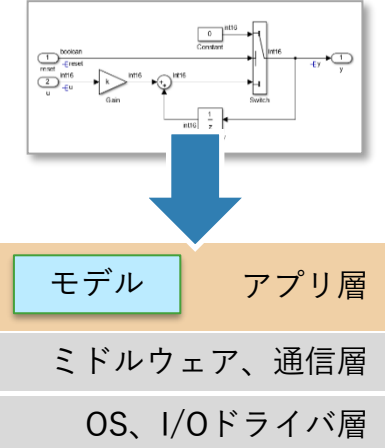
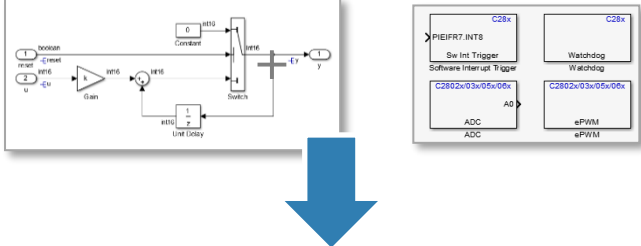


Task / Module	Throughput (uSec)	
	Model	Hand-Code
Current Magnitude and Phase Process	1.42	1.31
ABC to dq0 Frame Transformation	0.76	0.52
Resolver Harmonic Learn	0.48	0.22
Angle Position Determination	0.93	0.84
PI-Current Regulator	7.62	7.51
Torque Mode	4.82	4.72
dq0 Rotating to Stationary Frame Transformation	0.94	0.82
Complete 100 uSec Task	65.37	63.83

計算速度が同等

※MathWorks Automotive Conference Michigan 2015

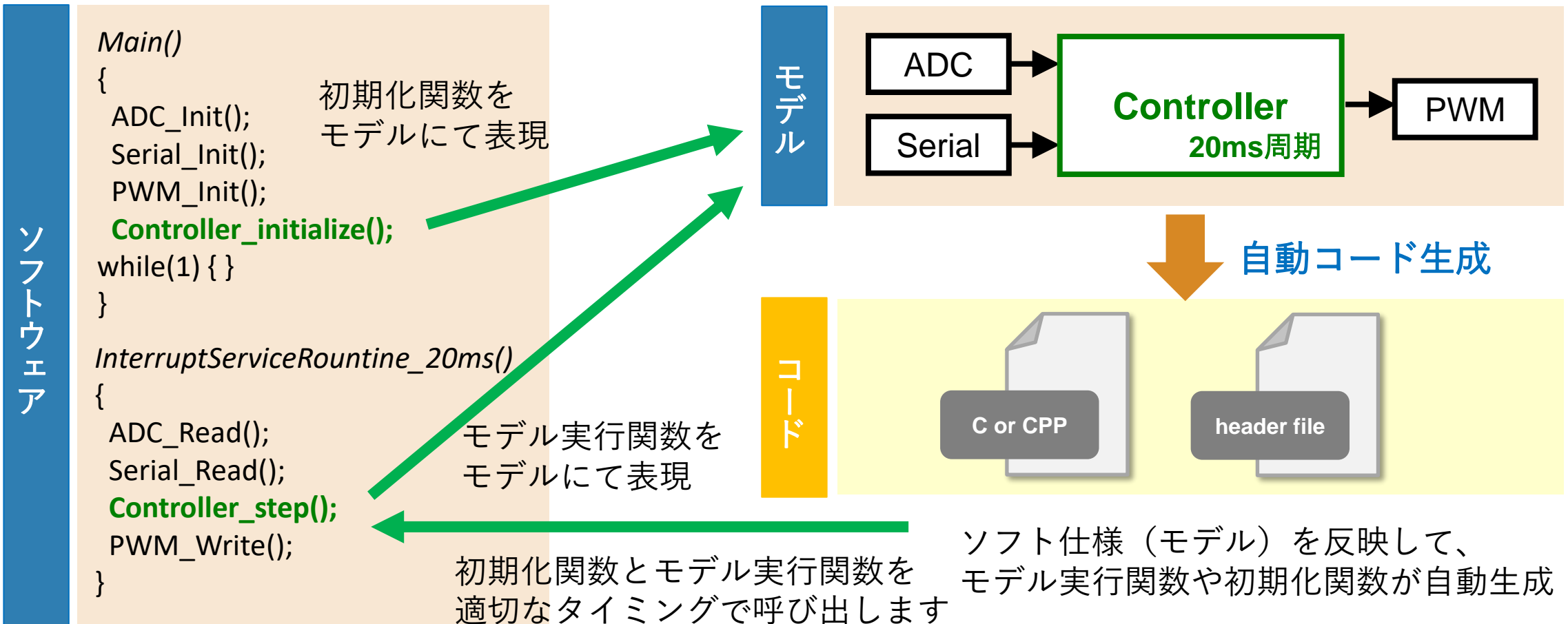
Embedded Coder 生成コードの利用タイプ

	ソフトコンポーネント型(基本)	モデルビルド型(応用)
利用イメージ	<p>アプリの一部をモデル化、生成されたコードを既存ソフトに組み込む</p> 	<p>MCU/DSP用ブロックライブラリ利用、モデルを直接実装する</p>  <p>Target MCU</p>
主な用途	既存ソフトとの差分開発 HW/OS非依存アプリの開発	MCU/DSPを用いた制御実験 (On Target RCP)
特徴	標準ブロックのみ使用	専用ブロックライブラリ*1を一部使用
I / O	グローバル変数 / 引数	AD/DA, PWM, CAN等のI/Oブロック
注意点	ビルド・結合は手動で行う必要アリ	対応MCU/DSP限定、HW知識も必要
対応MCU/DSP	生成コードはANSI/ISO-C準拠なので ほぼ全てのMCU/DSPに対応可能	TI C2000, STMicro STM32F4 等 (MathWorksまたはベンダから提供)

*1: [コード組み込み時に役立つサポートパッケージ](#)

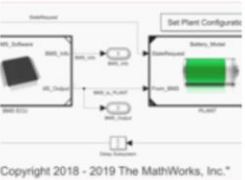
ソフトコンポーネント型の利用イメージ

- データ受け渡しはグローバル変数 or 引数
- 割り込みやポーリング処理でモデル実行関数を呼び出します



バッテリーマネジメントシステム（BMS）のレファレンス設計例

- レファレンス設計例に含まれるBMSアルゴリズムの例：
 - 拡張カルマンフィルタを用いたState of Charge推定
 - 受動バッテリーセルバランシング
 - フォールト管理 等



Design and Test Lithium Ion Battery Management Algorithms

バージョン 2.0.2 (35.1 MB) 作成者: Chirag **STAFF**

This example project can be used as a reference design to get started with designing Battery Management System with MATLAB and Simulink.

★★★★★ (19)
ダウンロード: 14.5K ⓘ
更新 2023/1/9
[ライセンスの表示](#)

[+ フォロー](#) [ダウンロード](#)

概要

関数

モデル

例

バージョン履歴

レビュー (19)

ディスカッション (54)

This example project can be used as a reference design to get started with designing Lithium Ion Battery Management System (BMS) with MATLAB and Simulink.

Project includes Simulink models for BMS Algorithms such as:

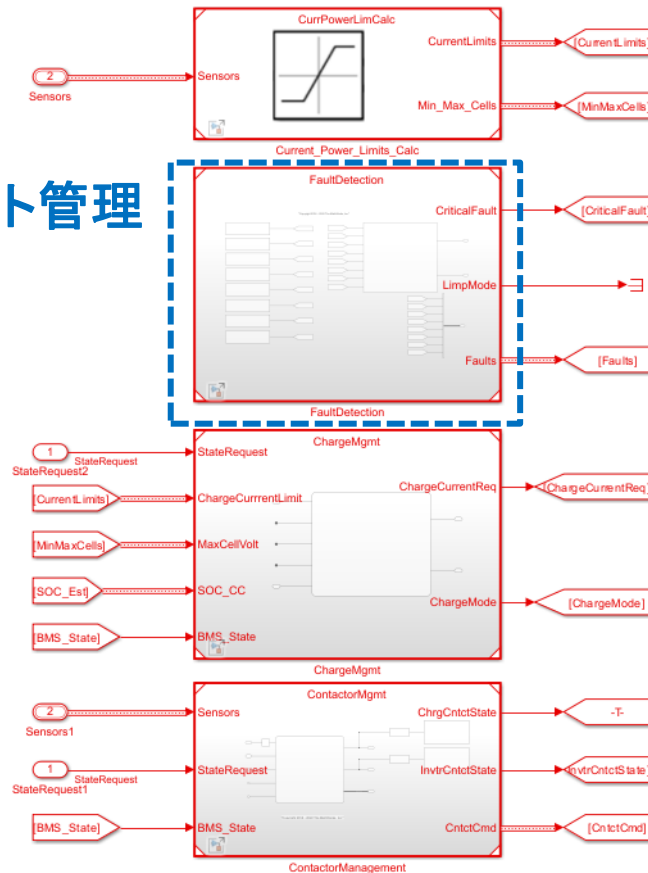
1. State of Charge estimation using Extended Kalman Filter, Unscented Kalman Filter
2. Passive Battery Cell Balancing
3. State Machine for Pre-charging and Contactor Management
4. Fault Management - Over/Under Voltage, Over Current, Over Temperature etc.
5. Charge and Discharge Current Limit Calculations

必須

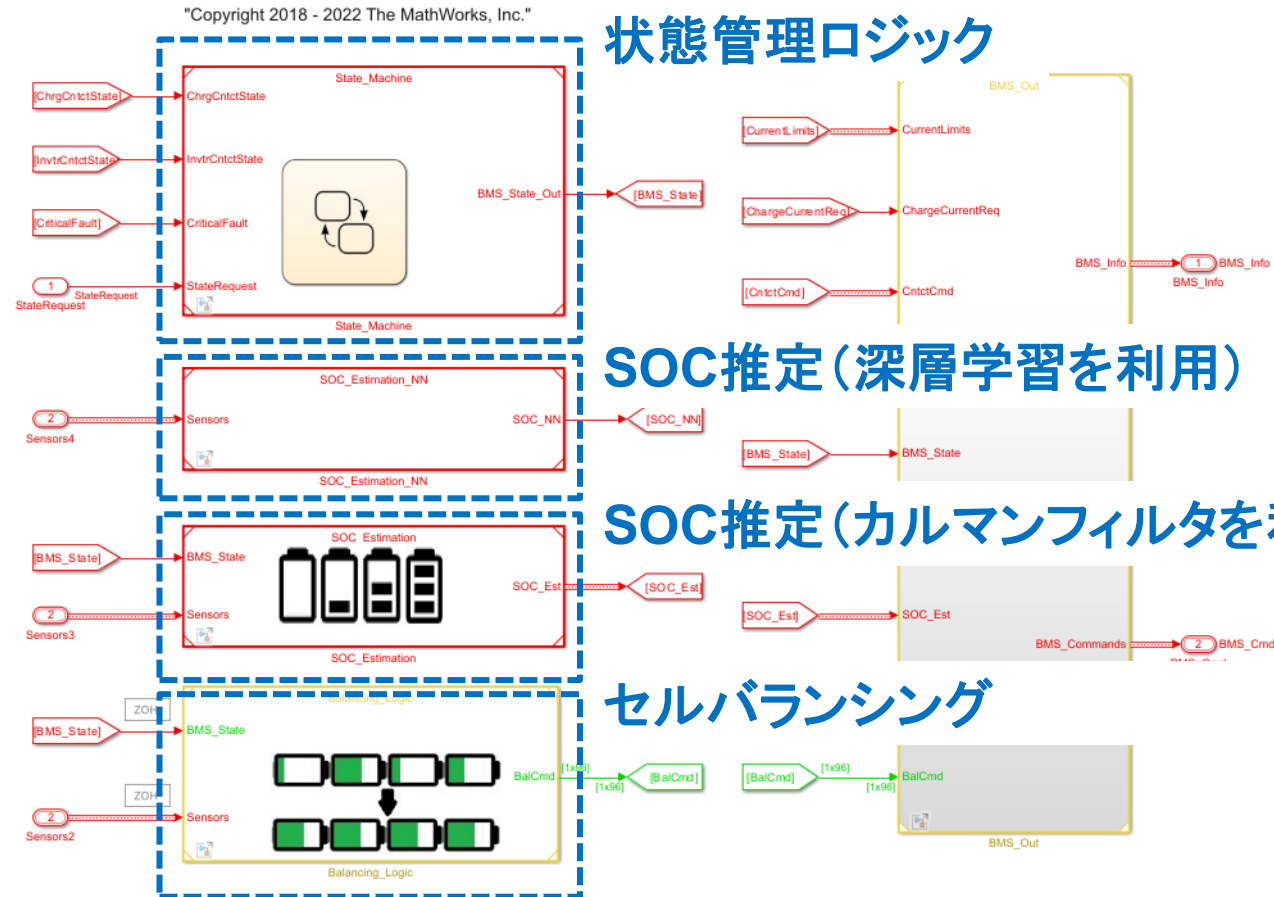
- MATLAB
- Simulink
- Control System Toolbox
- Requirements Toolbox
- Simscape
- Simscape Electrical
- Simulink Design Optimization
- Simulink Test
- Stateflow

バッテリーマネジメントシステム (BMS) アルゴリズムの例

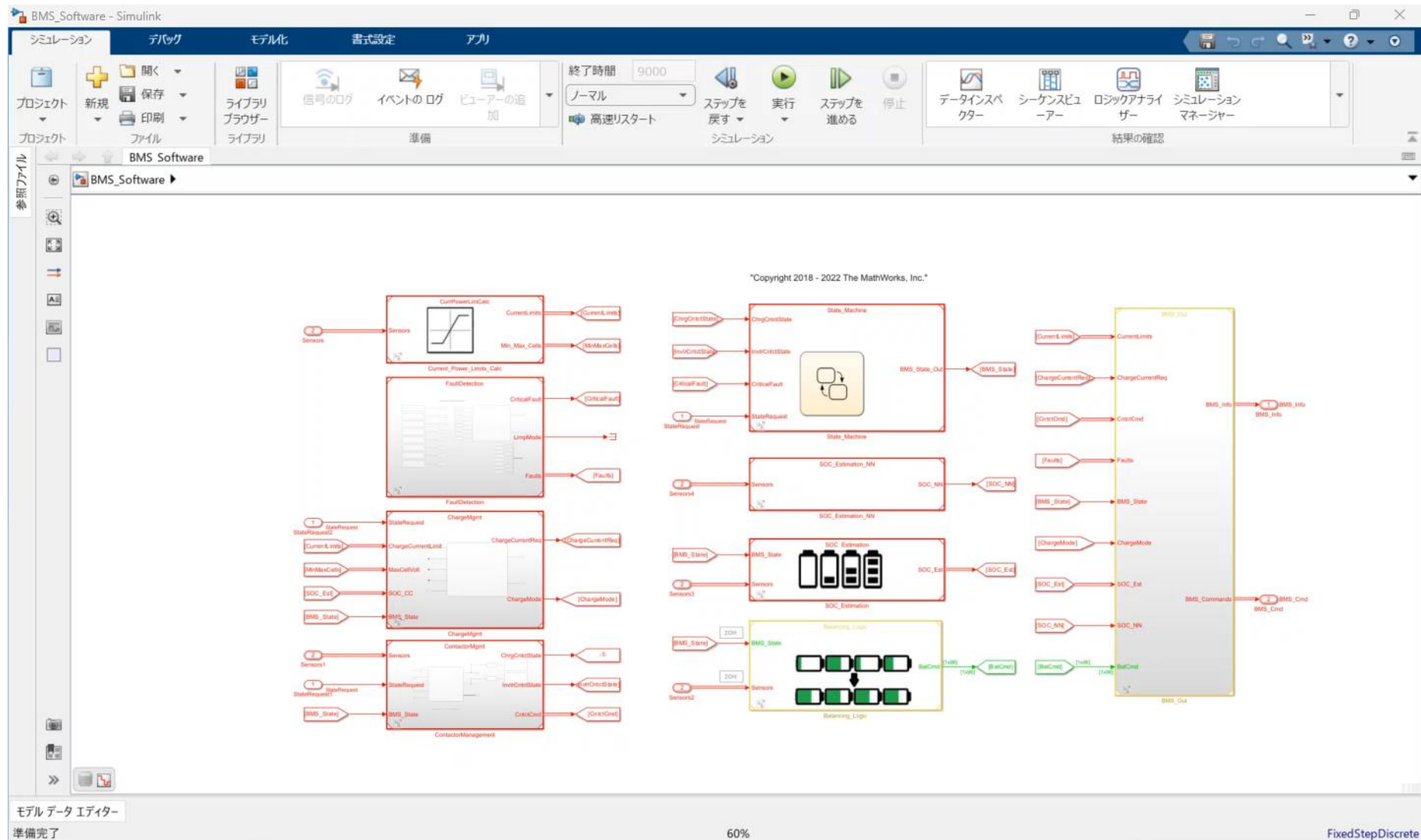
フォールト管理



"Copyright 2018 - 2022 The MathWorks, Inc."



デモ動画*1



*1：本デモ動画はレファレンス設計を改変し作成

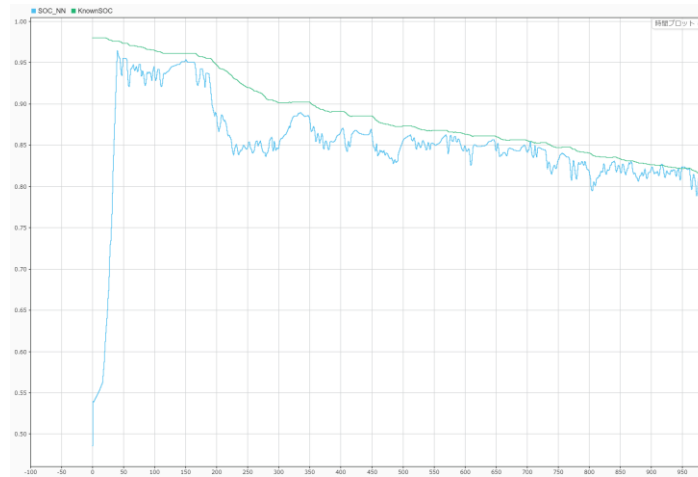
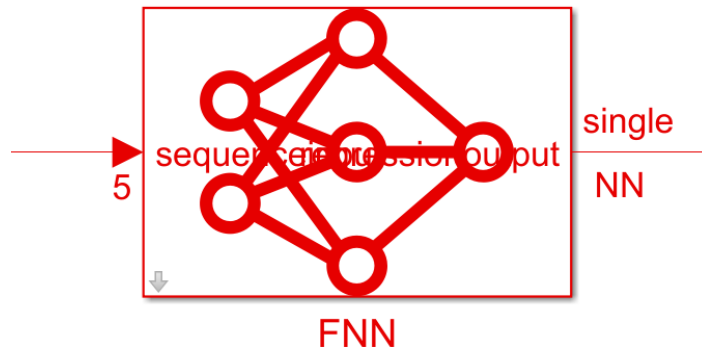
【新機能】 深層学習アルゴリズムのコード生成

■ 用途

- 深層学習で高精度なSOC推定を実現、実装したい
(実現例：BMSレファレンスモデル内の SOC_Estimation_NN.slx)

■ 手段

- コード生成*1に対応したDeep Learning Toolbox 深層学習用ブロックを利用



コード
自動生成



図：深層学習用Simulinkブロック

図：深層学習によるSOC推定のイメージ

*1：[Simulinkを利用した深層学習](#), [Simulink アプリケーションからの深層学習コードの生成](#)

アジェンダ

- 背景
- マイコン向け量産コード生成の紹介
 - コード生成ツール Embedded Coderの概要
 - バッテリーマネジメントシステム開発への適用例
- シミュレータによるコントローラの機能評価の紹介 (HILS)
 - MathWorksのRCP/HILSソリューションの概要
 - バッテリーマネジメントシステム開発への適用例

HILS: Hardware-In-the-Loop Simulation

- 実機をシミュレータで模擬し、量産/試作コントローラの機能評価を実施

実機による機能・性能評価：



量産/試作HW



実際の制御対象（実機）



実機の振る舞いをシミュレータで模擬

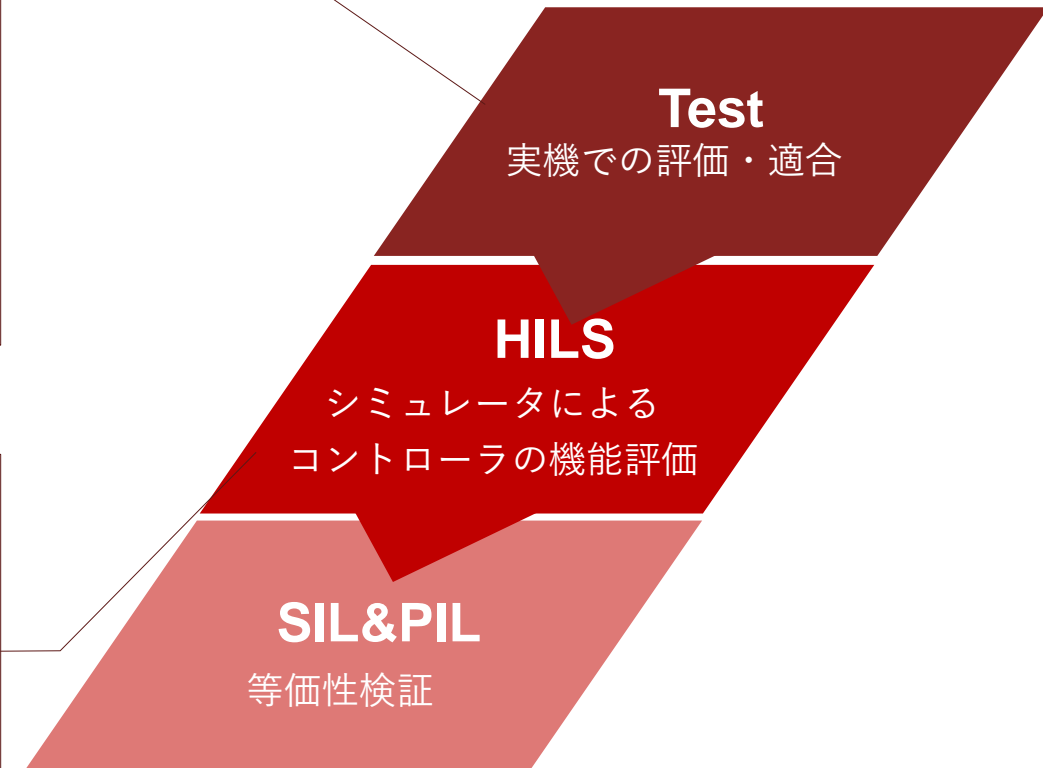
HILSによる機能・性能評価：



量産/試作HW



HILシミュレータ



HILS導入のメリット



任意の試験条件の素早い再現



再現性の高い繰り返し試験



繰り返し試験の自動化



実際の設備の占有時間を削減
(現場に行かずに最大限のデバッグ)



実物を壊さずに故障モードを再現

HILSテスト環境のイメージ：



量産/試作HW



通信/I/O



HILシミュレータ

モデルのビルド & ダウンロード



信号のモニタ・ロギング



実機振る舞いのモデリングツール
HILシミュレータ連携ツール

HILSテスト環境 Simulink Real-Time ・ Speedgoat

- HILSテスト環境
 - Simulink Real-Time：HILS用ユーティリティを提供するオプション製品
 - Speedgoat：HILS環境用のハードウェア
- 特徴
 - MATLAB/Simulinkからクリック1つでリアルタイムテストに移行可能
 - Speedgoat用I/Oブロックで効率的にHILS環境が構築可能
 - マルチコアCPU/FPGAを活用した高速演算

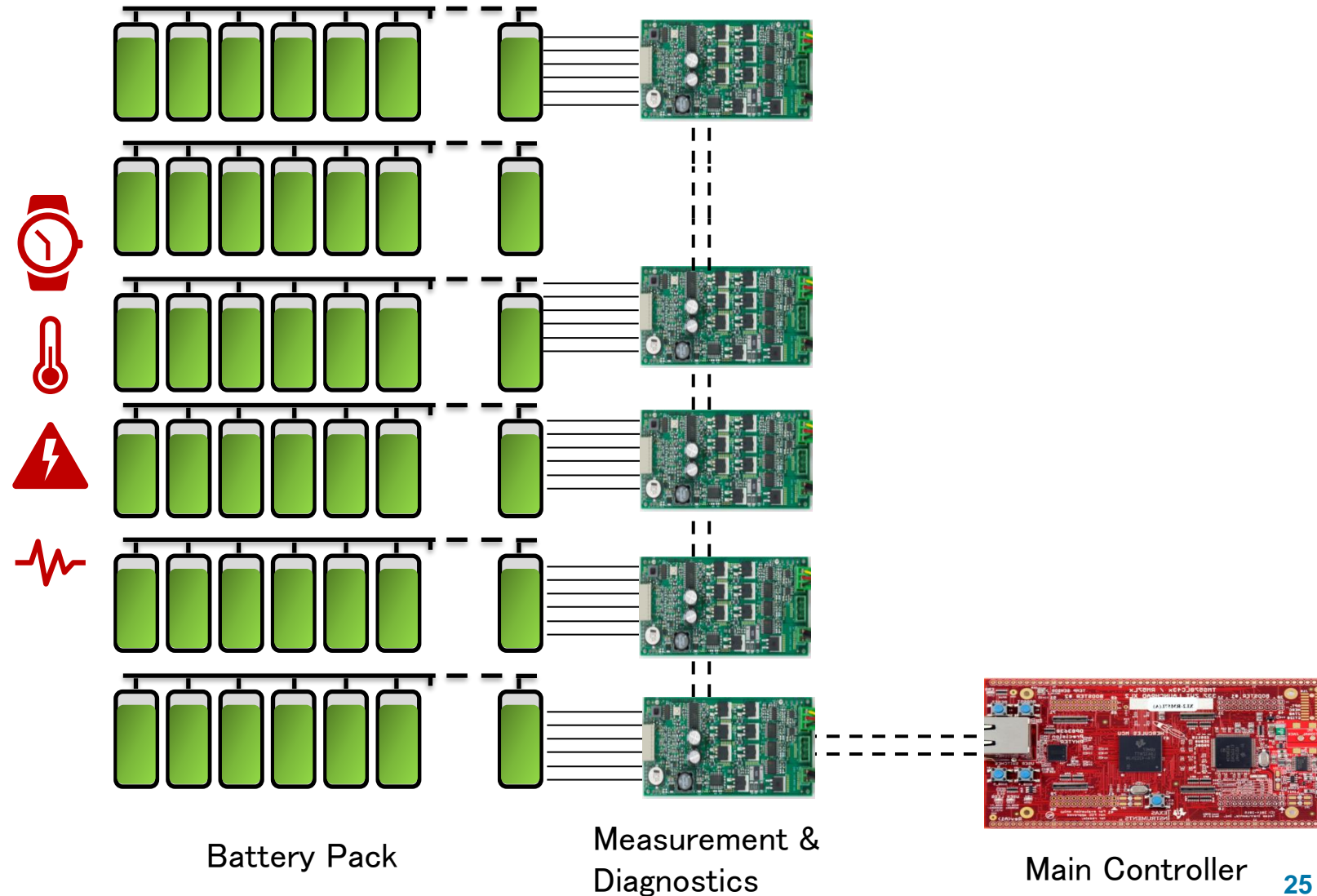
HILSテスト環境の構築例：



バッテリーマネジメントシステム (BMS) に対するHILS構築の利点

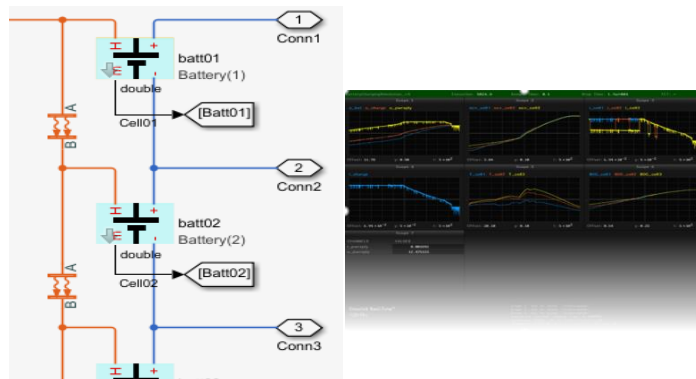
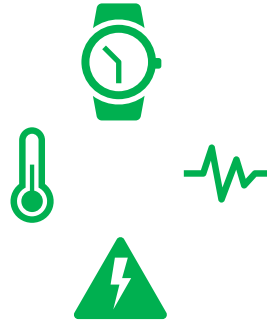
BMS実機テストの課題：

- ✓ テスト準備のテストサイクルの長さ
- ✓ 困難な結果の再現
- ✓ 危険な故障条件下でのテスト
- ✓ テストの自動化が限定的



バッテリーマネジメントシステム (BMS) に対するHILS構築の利点

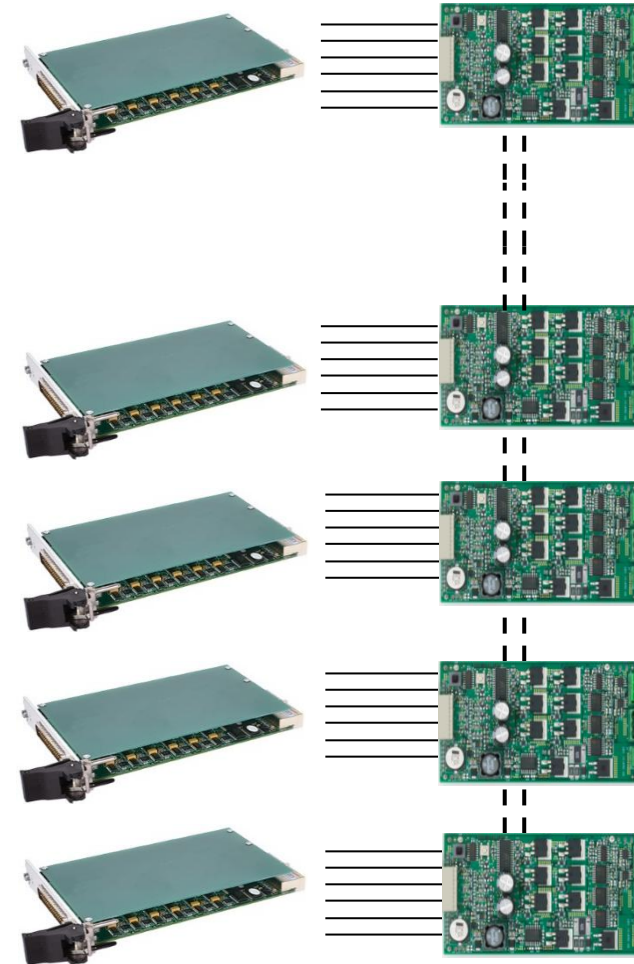
- ✓ 任意の試験条件の素早い再現
- ✓ 試験条件の高い再現性
- ✓ 故障モードの動作確認



自動コード生成



Wiring and Signal Conditioning



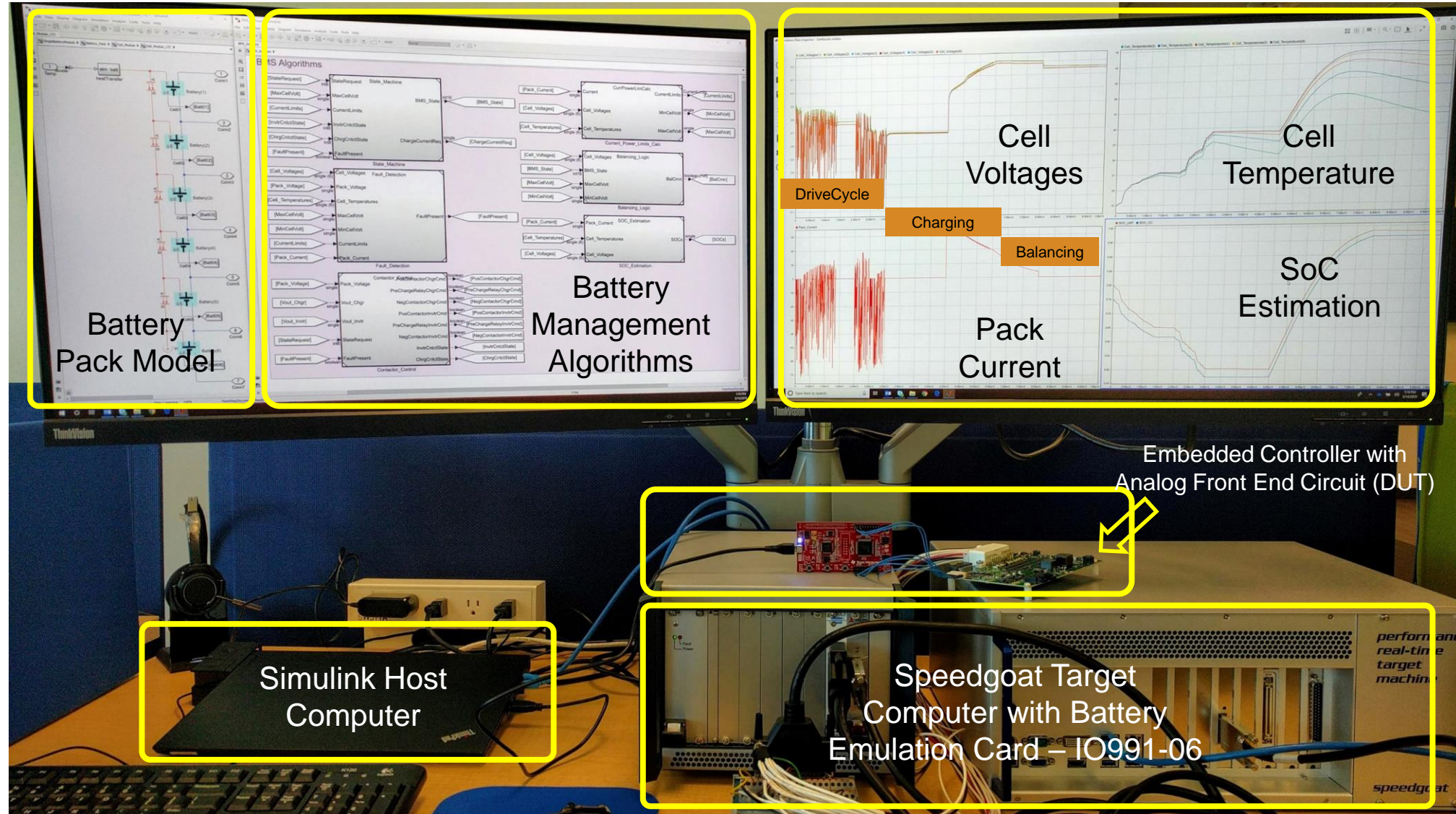
Battery Emulation

Measurement &
Diagnostics



Main Controller

BMSに対するHILS構築例



まとめ

まとめ

- マイコン向け量産コード生成の紹介
 - モデルから自動コード生成、組み込みまでの概略
 - BMSレファレンス設計におけるデモ

- HILSの紹介
 - シミュレーションを活用したHILSテスト手法の概略
 - BMSテストにおけるHILS活用例

導入に向けたお役立ち情報

■ Embedded Coder

- 製品ドキュメント付属の[Embedded Coder入門](#)をお試してください。
- ツール導入に向けた評価時には弊社営業にお声がけください。
 - 評価時に参考となる資料なども適宜ご紹介致します。
- スムーズな導入のために、弊社トレーニング受講もおすすめします。
 - [Embedded Coderによる量産向けコード生成](#)

■ Simulink Real-Time/Speedgoat

- 是非とも下記セミナー動画をご参照ください。
 - [ハードウェアインザループシミュレーション\(HILS\)による量産/試作コントローラのテスト](#)
- ツール導入に向けた検討時には弊社営業にお声がけください。
 - 詳細に当該製品を紹介致します。



Accelerating the pace of engineering and science

© 2023 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.