

Simulink Test™ Tutorial

- V1.0 -

MathWorks Japan

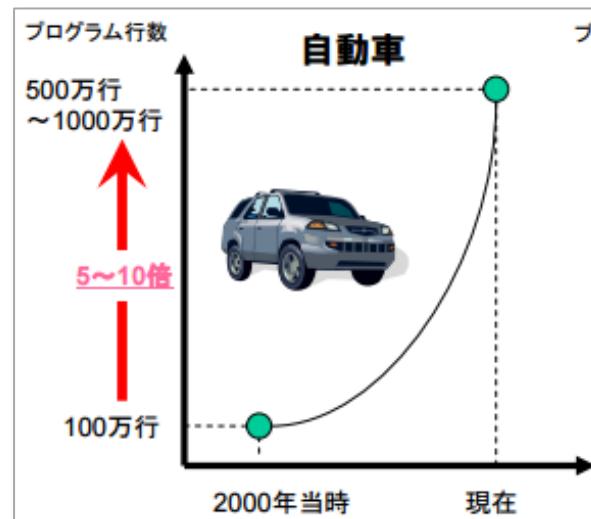
コンテンツ

1. Simulink Test について
2. Simulink Test の基本操作の習得
 - テストハーネス作成
 - シーケンシャルなテスト作成
 - 定型テストの自動化
3. まとめ & 参考情報

モデルベースデザイン/開発（MBD）が量産制御ソフト開発に求められる背景

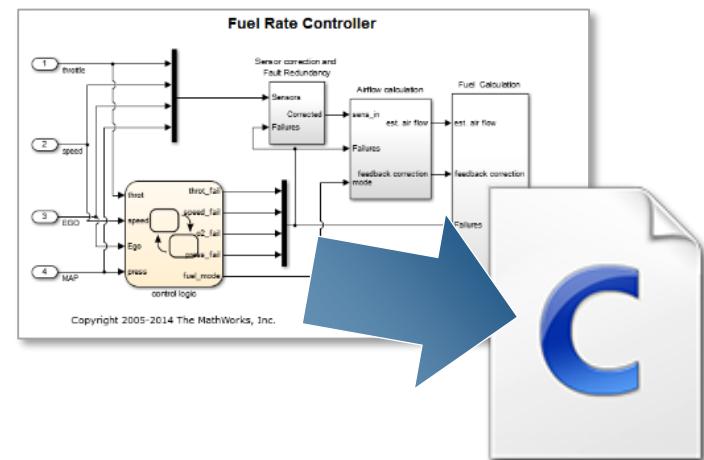
課題

- ・ソフト規模の巨大化
- ・検証項目の増加
- ・開発期間の維持・短縮



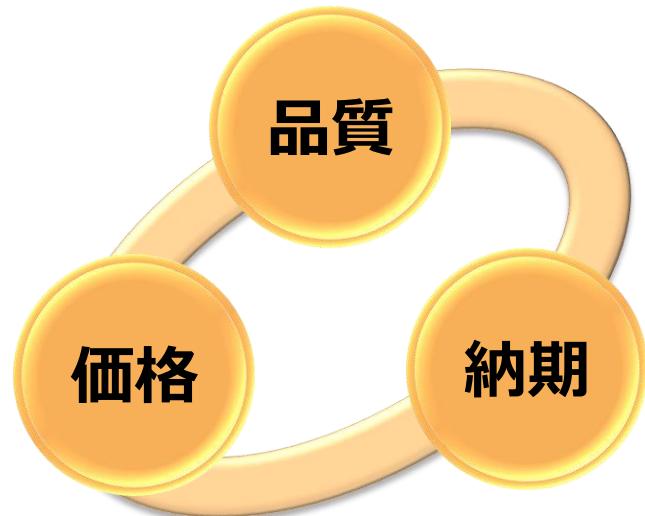
解決策

- ・モデル＆シミュレーションを通じた設計・**検証の前倒し**
- ・コード自動生成ツールや検証**ツールを用いた省力化**



効果

- ・早期の制御仕様確定
- ・開発効率向上
- ・ソフト信頼性向上

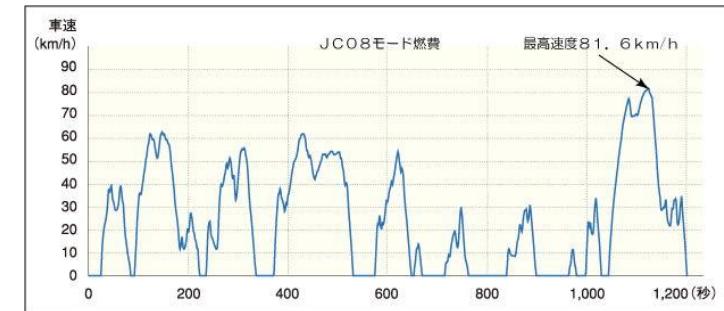
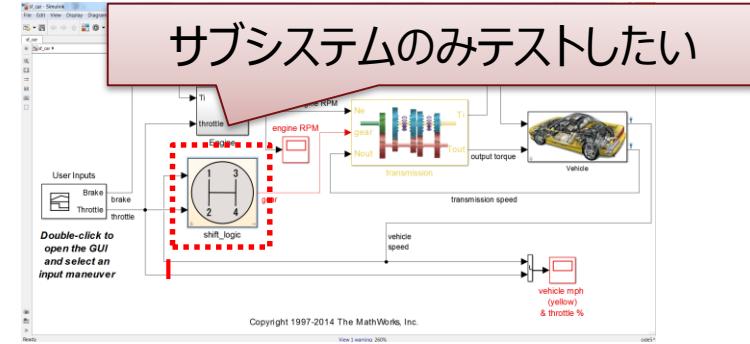


※ 2011 経済産業省資料より引用

MATLAB&SIMULINK®

シミュレーションテストに対するニーズは？

- **単体・統合テストをシームレスに行いたい**
 - サブシステム単位でテストしたい
(できれば別モデルに切り出してテストしたくない)
 - 複数のテストモデルを効率的に試したい
- **複雑なテストパターンを簡単に作成したい**
 - タイミングチャートを手描きで作成するのは大変
 - 入力値に応じてテストパターンを切り替えたい
(動的タイミングチャート)
- **複数テストを自動実行して合否レポートをみたい**
 - 入出力読み込み→テスト一括実行→レポート作成
 - 内製ツールで対応するとツールの管理・維持が大変

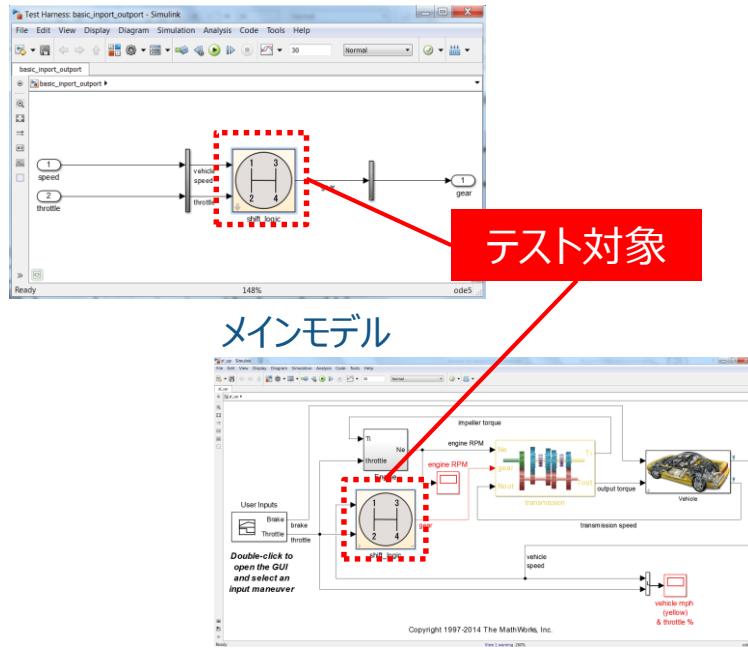


Simulink Test はテスト・検証業務の効率化をサポートします

テストハーネス作成

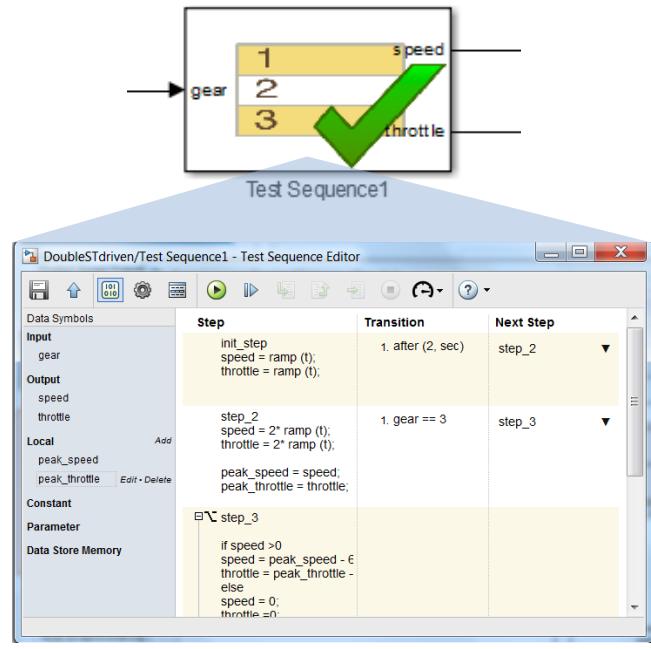
モデル全体・サブシステム単体・参照モデルに対して複数のテストハーネス(テスト用モデル)を簡単定義

テストハーネス



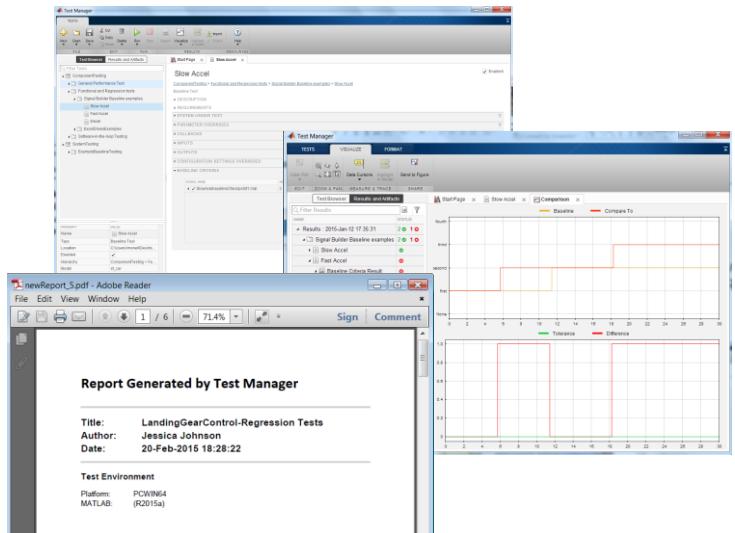
シーケンシャルなテスト作成

テストシーケンスブロックにより
状態遷移表ベースに複雑な入力
パターンを簡潔表現



定型テストの自動化

テストマネージャーによる
MIL/SIL/PILテストの
自動化・レポート作成・テスト管理



(テスト対象としてモデル全体
or テストハーネスを指定可能)

コンテンツ

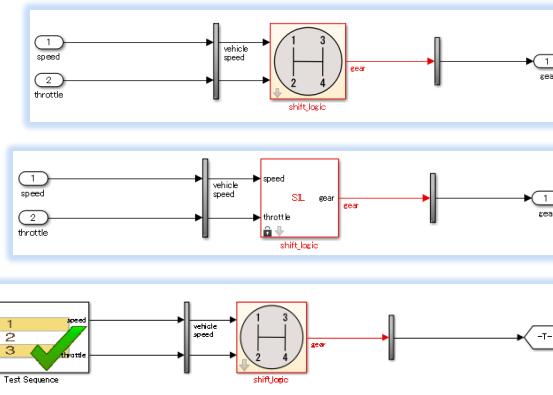
1. Simulink Test について
2. Simulink Test の基本操作の習得
 - テストハーネス作成
 - シーケンシャルなテスト作成
 - 定型テストの自動化
3. まとめ & 参考情報

テストハーネス機能

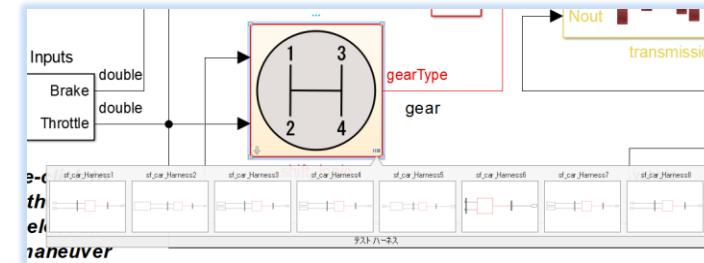
単体・統合テストをシームレスにします

- モデル全体・サブシステム単体・参照モデル用テストハーネスマルを作成・関連付け
- 複数テストハーネスを作成・管理可能
- 様々なテスト入出力ブロックを設定可能
- テストハーネスはメインモデルに同期、修正内容を自動で反映
- Data Store Memoryに対応
- SIL/PILに対応

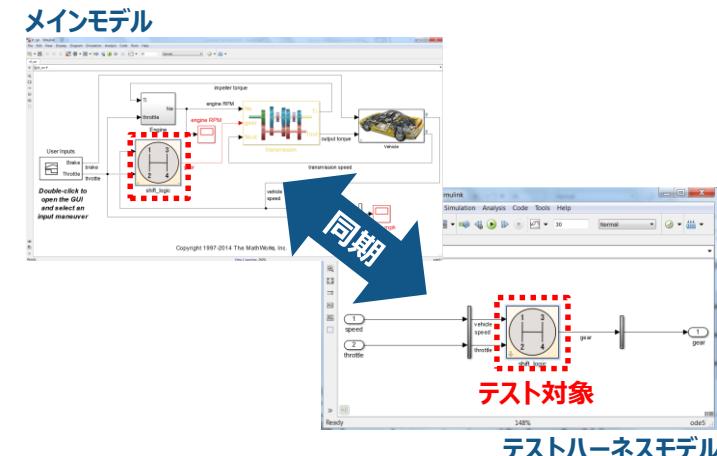
簡易なテストハーネスの作成



テストハーネスの関連付け・管理



モデル/ハーネスの同期

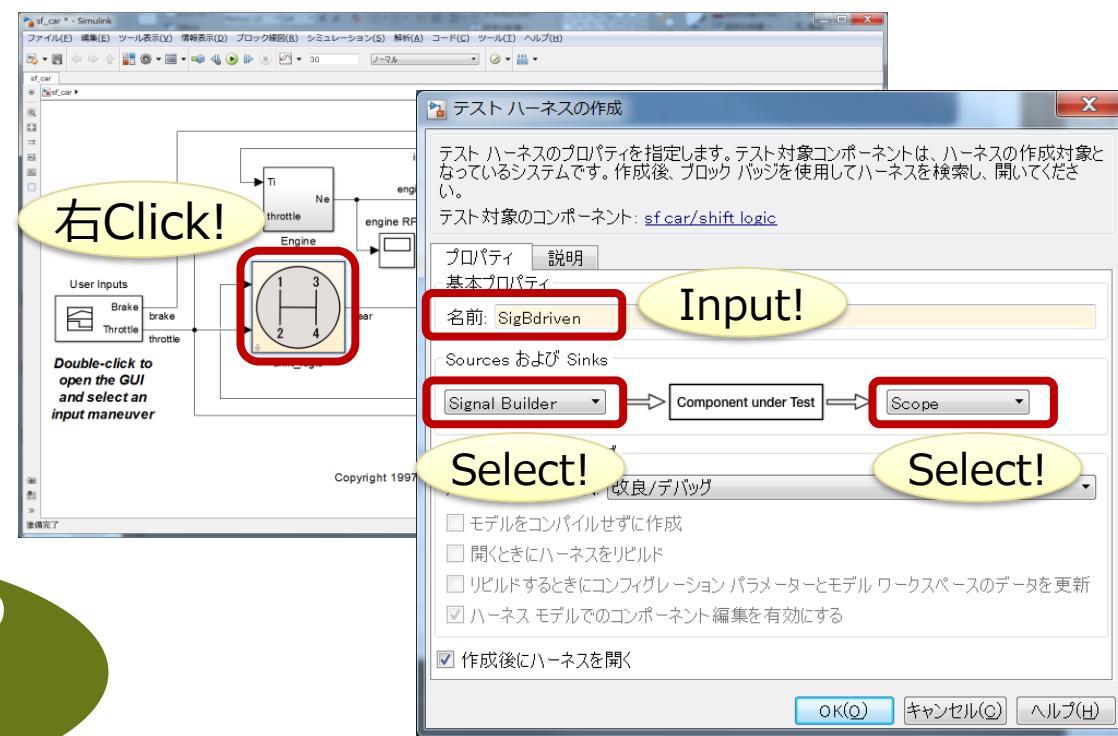
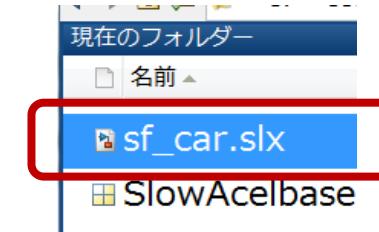


演習 1) テストハーネス作成

1. `sf_car.slx` を開きます。
 - MATLAB から `[sf_car.slx]` をダブルクリックして開きます。

2. `shift_logic` 用のテストハーネスを作成します。
 - `[shift_logic]` → 右クリック → [テスト ハーネス] → [テスト ハーネス (`shift_logic`) の作成]
 - [名前] → `[SigBdriven]` を入力します。
 - [Sources および Sinks]
 - Sources に `[Signal Builder]` を選択し
 - Sinks に `[Scope]` を選択します。
 - [OK] をクリックします。

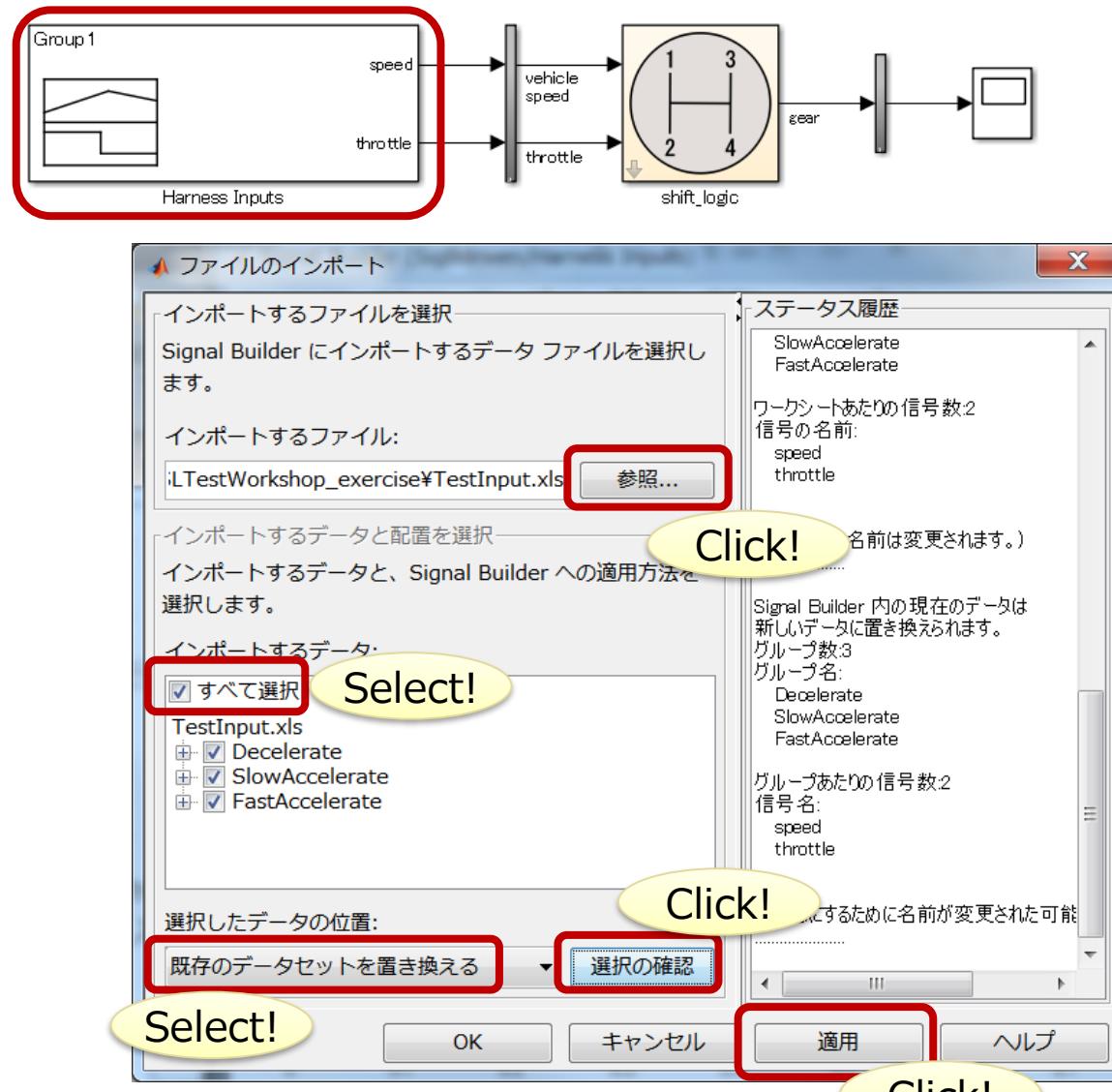
メリット① : テスト用の
ハーネスマルを簡単
に作成可能



演習 1) テストハーネス作成

3. テスト入力を作成します。

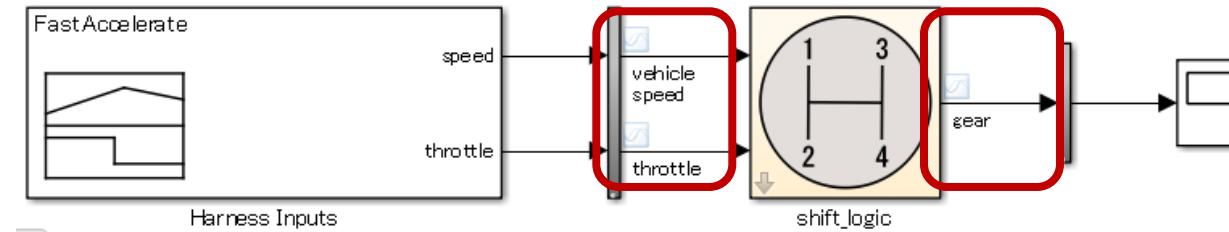
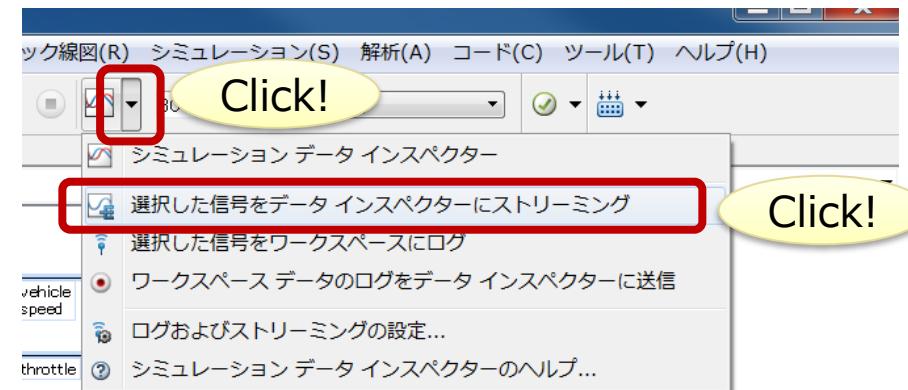
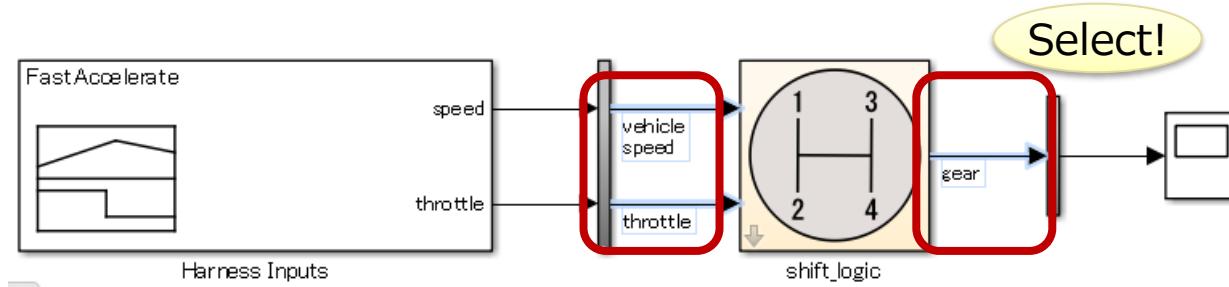
- [Harness Inputs] をダブルクリックして開きます。
- [インポートするファイル] → [参照] → [TestInput.xls] を選んで外部Excelファイルからテスト入力信号をSignal Builderにインポートします。
- [インポートするデータ] → [すべて選択] でExcel中に入っている3つのSheetをすべてインポートします。
- [選択したデータの位置] → [既存のデータセットを置き換える] → [選択の確認] → [適用]
- [OK] をクリックし、データがSignal Builderにインポートされていることを確認します。



演習 1) テストハーネス作成

4. ストリーミング信号を指定します。

- [vehicle speed]、[throttle]、[gear] の信号線をマウスカーソルで選択します。
- ツールバーにある
[シミュレーション データ インスペクター]  の
プルダウンメニューから
[選択した信号をデータ インスペクターにスト
リーミング] を選択します。
- 選択した信号にストリーミングのマーク  が表示されます。



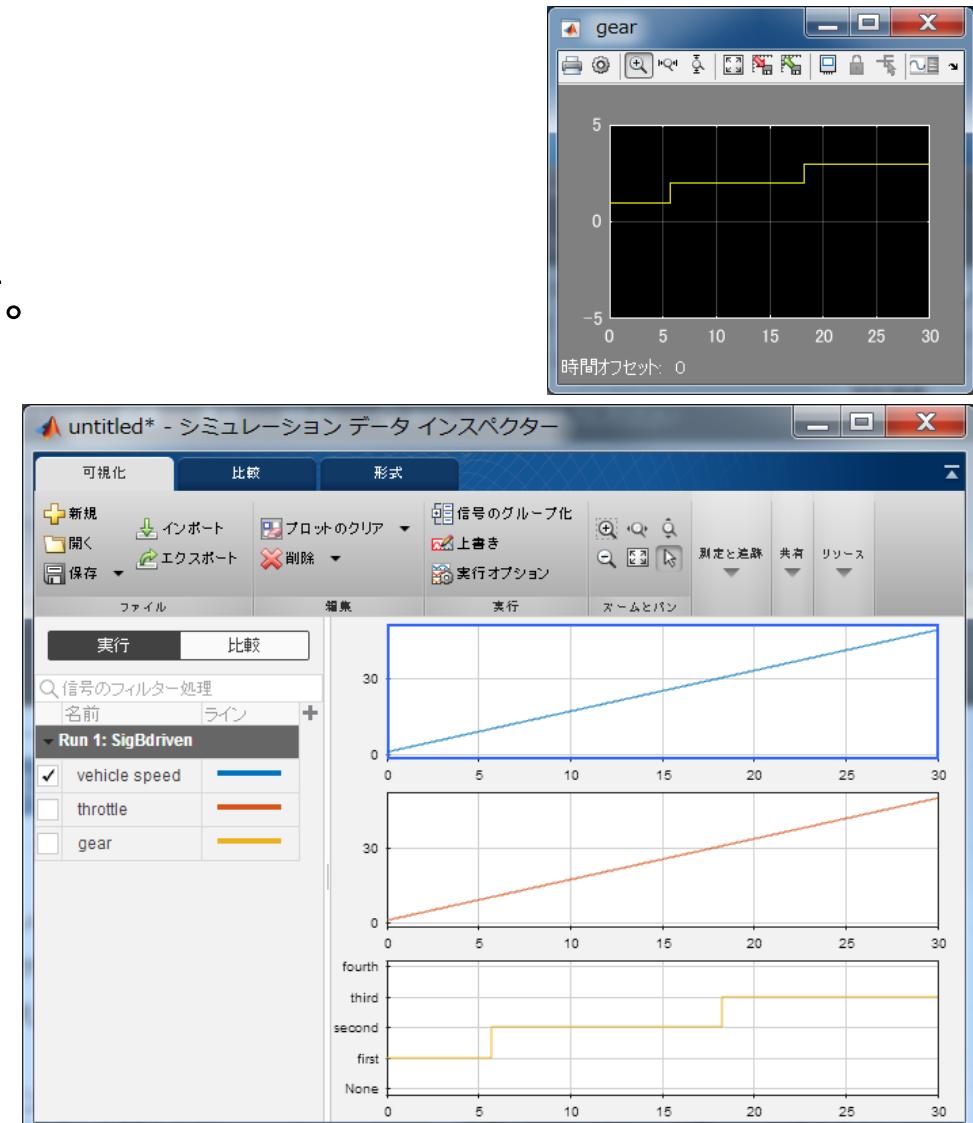
演習 1) テストハーネス作成

5. シミュレーションを実施し、結果を確認します。

- [実行] ボタン  をクリックし、シミュレーションを実施します。
- [Scope] をダブルクリックし、[gear] の信号線を確認します。

もしくは、

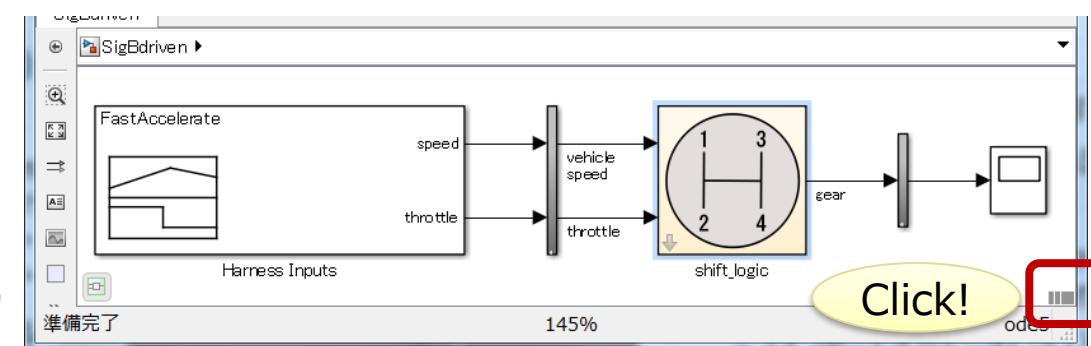
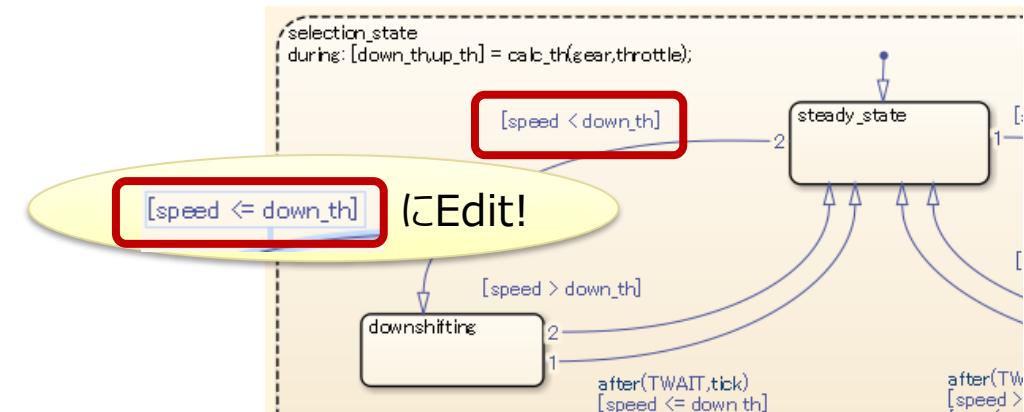
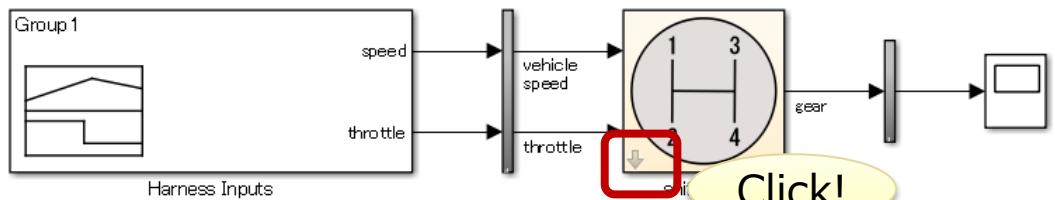
- ハイライトされている [シミュレーションデータ インスペクター]  をクリックし、データインスペクターを開きます。
- ステップ4で設定した [vehicle speed]、[throttle]、[gear] のストリーミング信号が左側に表示されます。
- 右側のグラフに、上から順番に [vehicle speed]、[throttle]、[gear] 信号をアサインし、シミュレーションの入出力を確認します。



演習 1) テストハーネス作成

6. テストハーネス内の[shift_logic]修正し、
メインモデルとの同期を確認します。
- [shift_logic] の左下にある矢印アイコンをクリックしてチャートを開きます。
 - [steady_state] から [downshifting] への遷移条件を [speed <= down_th] に編集します。
 - [SigBDriven] のトップ階層に戻り → モデルウィンドウの右下にある長方形のアイコンをクリックし、メインモデルに切り替えます。
 - メインモデル内の [shift_logic] を開き、修正が反映されていることを確認します。

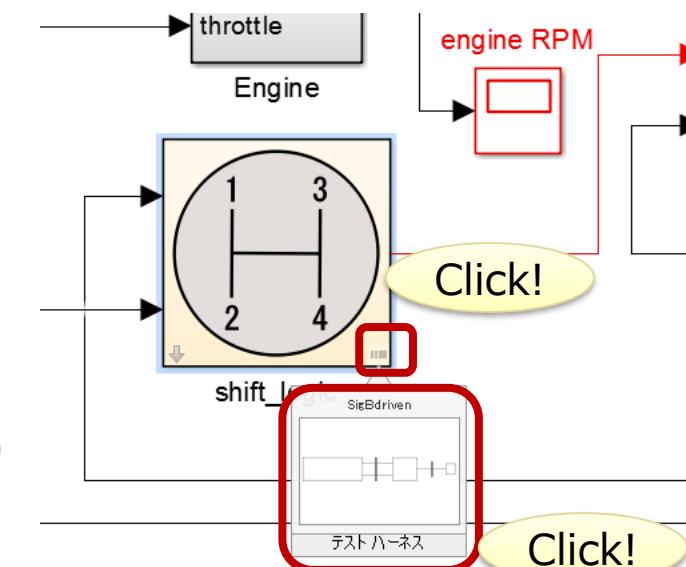
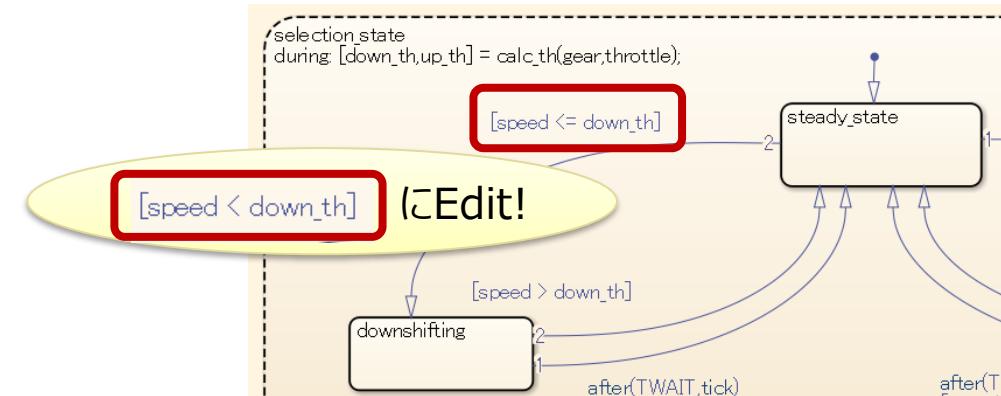
**メリット②：手修正による
修正の反映漏れを防ぐ
ことが可能**



演習 1) テストハーネス作成

7. メインモデル内の[shift_logic]修正し、テストハーネスとの同期を確認します。
 - メインモデルの [shift_logic] を開いたまま、[steady_state] から [downshifting] への遷移条件を [speed < down_th] に戻します。
 - [sf_car] のトップ階層に戻り → [shift_logic] の右下にある長方形のアイコンをクリックし、テストハーネス [SigBdriven] に切り替えます。
 - テストハーネス内の [shift_logic] を開き、修正が反映されていることを確認します。

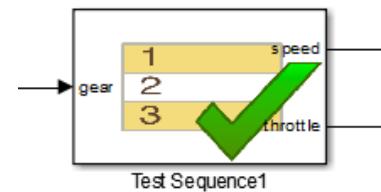
メリット③ : テストハーネスと
メインモデルが同じ.slxで保存
されているため、管理も楽



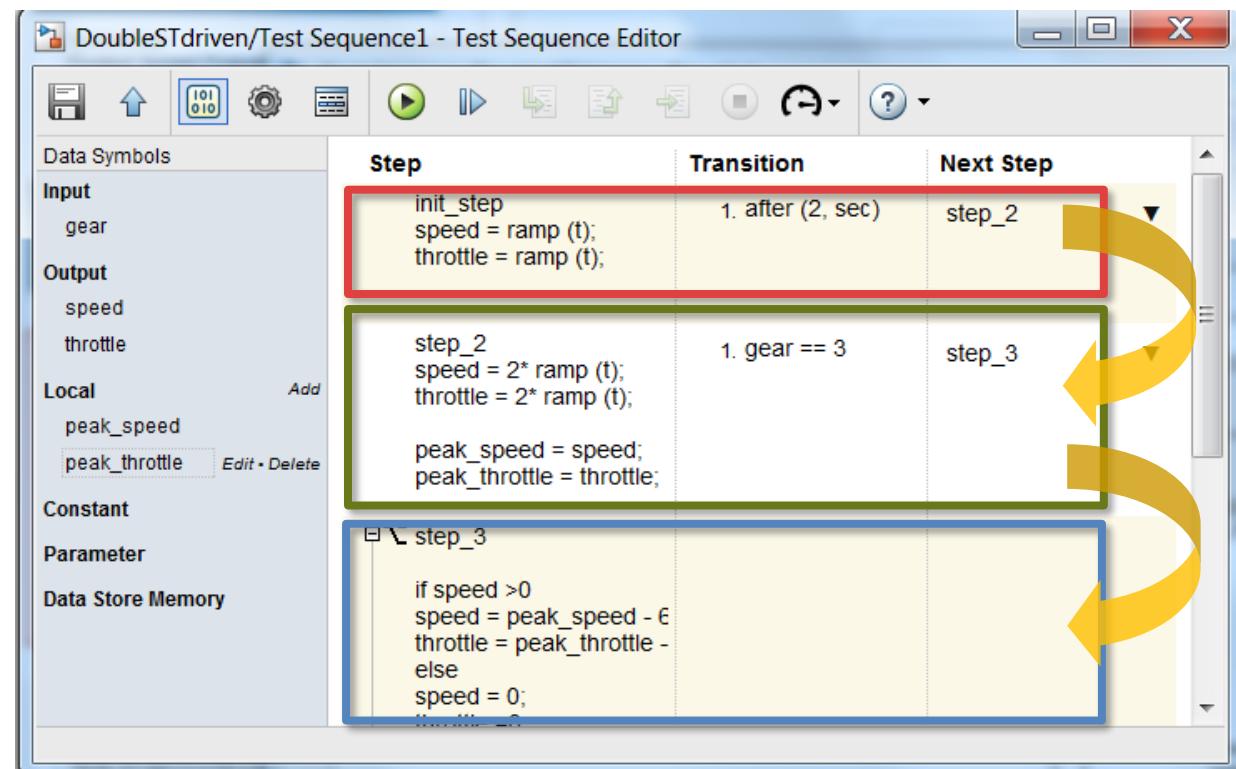
テストシーケンスブロック

複雑なテストパターンを簡単に作成できます

- 状態遷移表を用いて複雑なテストパターンを作成可能
- プラント出力や内部状態を入力として受け取って、テストパターンを切り替え可能（動的タイミングチャート）
- 診断(assert)挿入による信号チェック・不具合検証が可能



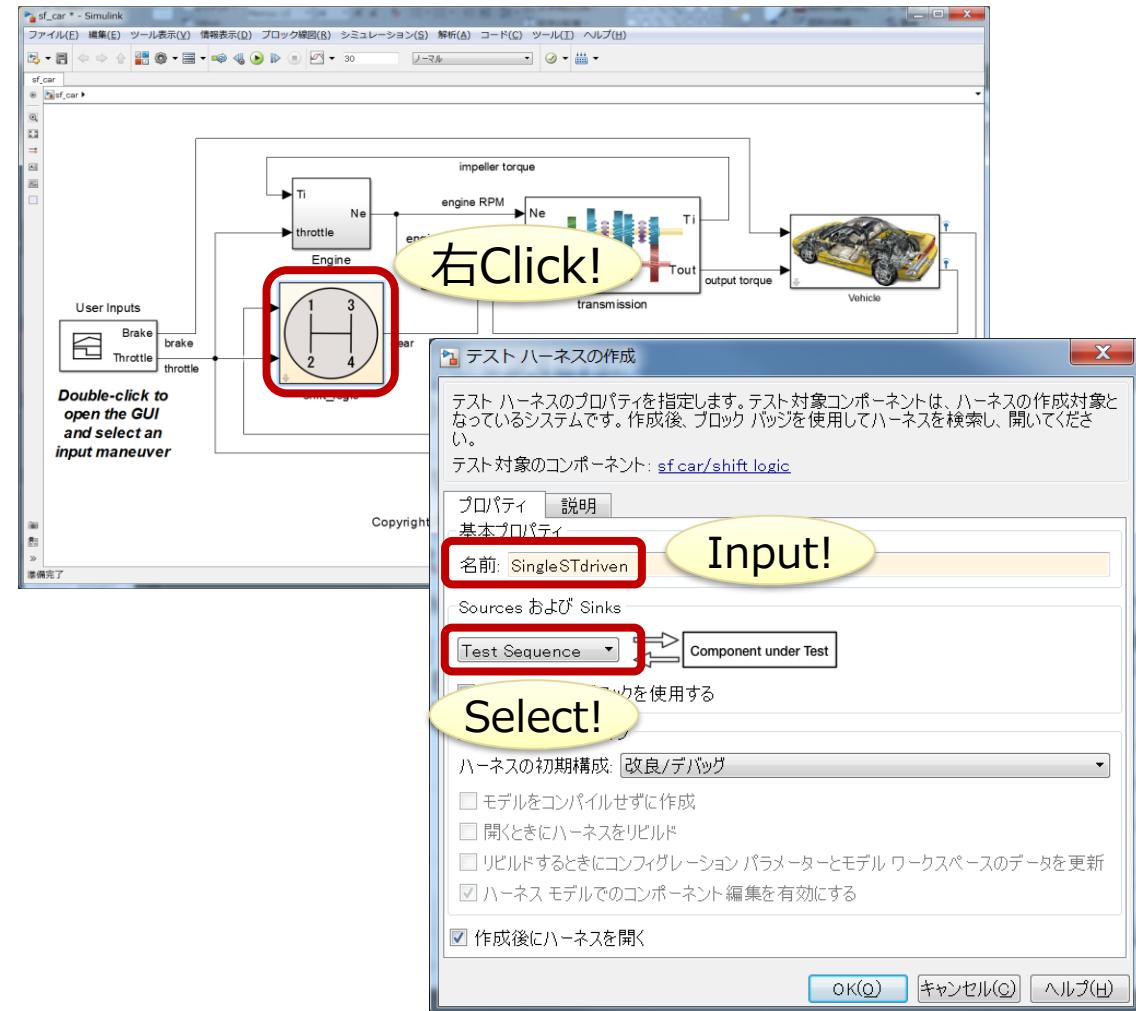
条件を満たすと次の行に遷移



演習 2) シーケンシャルなテスト作成

1. shift_logic 用のテストハーネスを作成します。

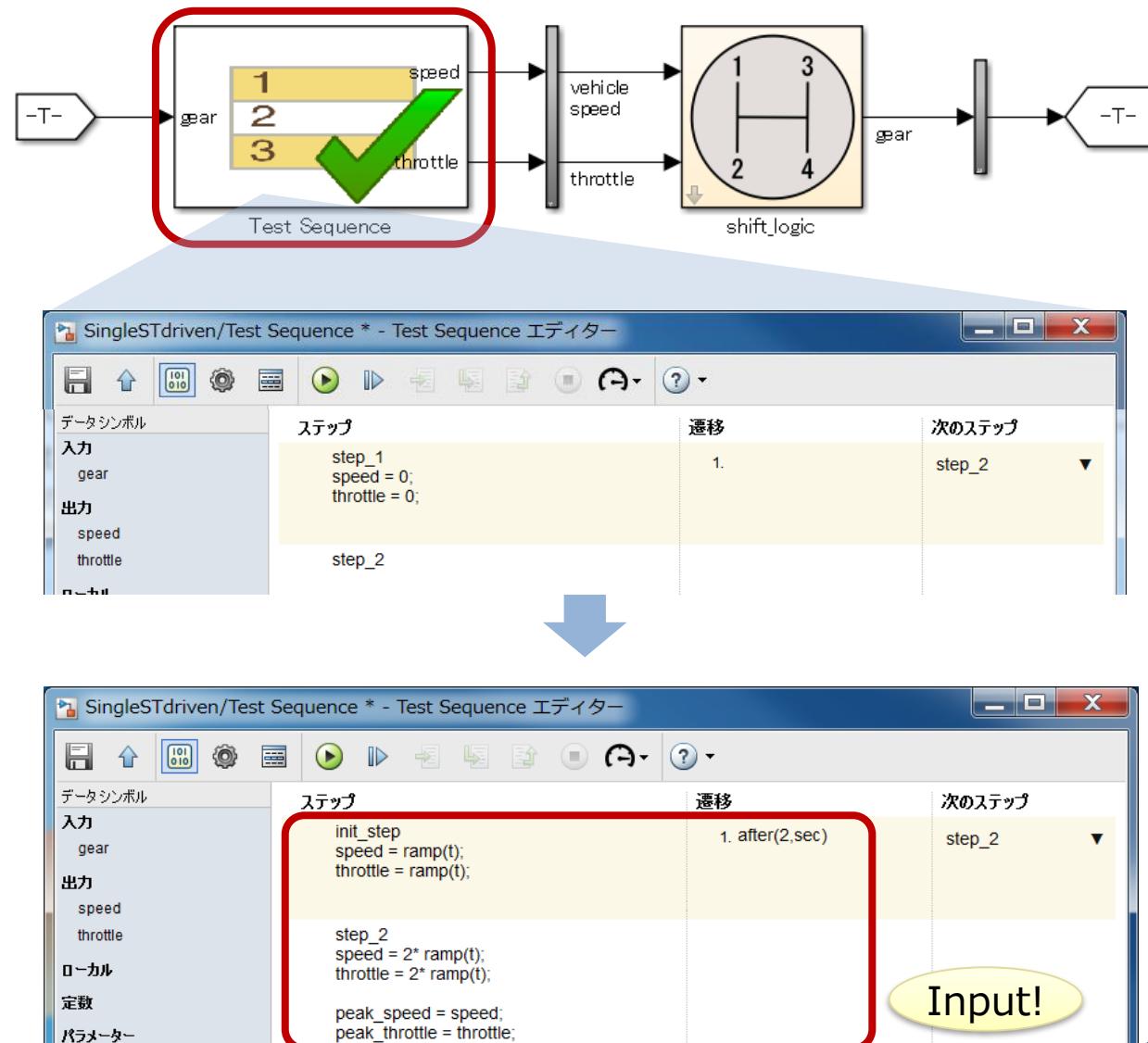
- [shift_logic] → 右クリック → [テスト ハーネス]
→ [テスト ハーネス (shift_logic) の作成]
- [名前] → [SingleSTdriven] を入力します。
- [Sources および Sinks]
→ Sources に [Test Sequence] を選択します。
- [OK] をクリックします。



演習 2) シーケンシャルなテスト作成

2. テスト入力を作成します – その1

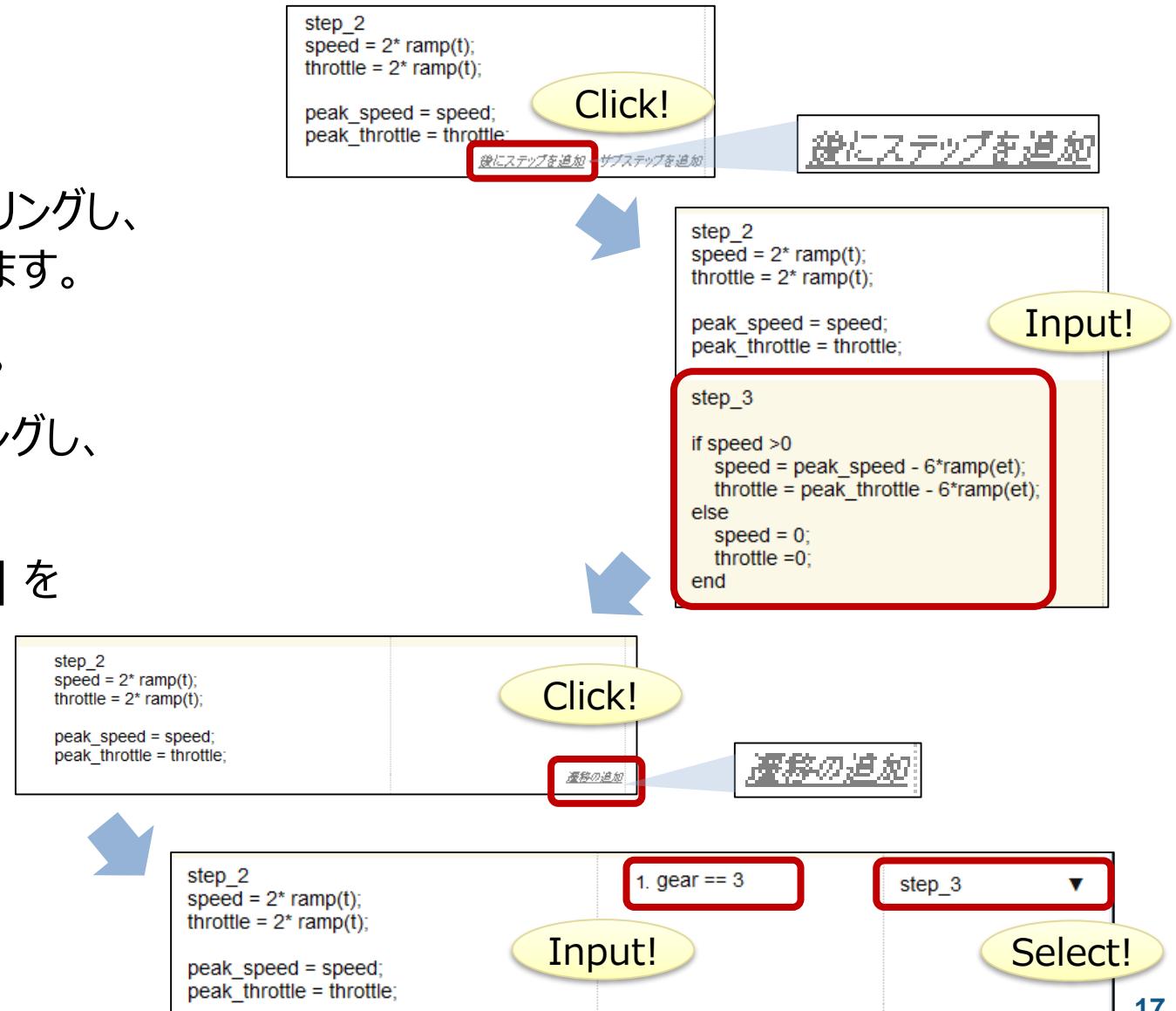
- [Test Sequence] ブロックをダブルクリックして、[Test Sequence エディター] を開きます。
- 右下の図のように [init_step] の作業や遷移条件と [step_2] の作業を定義します。



演習 2) シーケンシャルなテスト作成

2. テスト入力を作成します – その 2

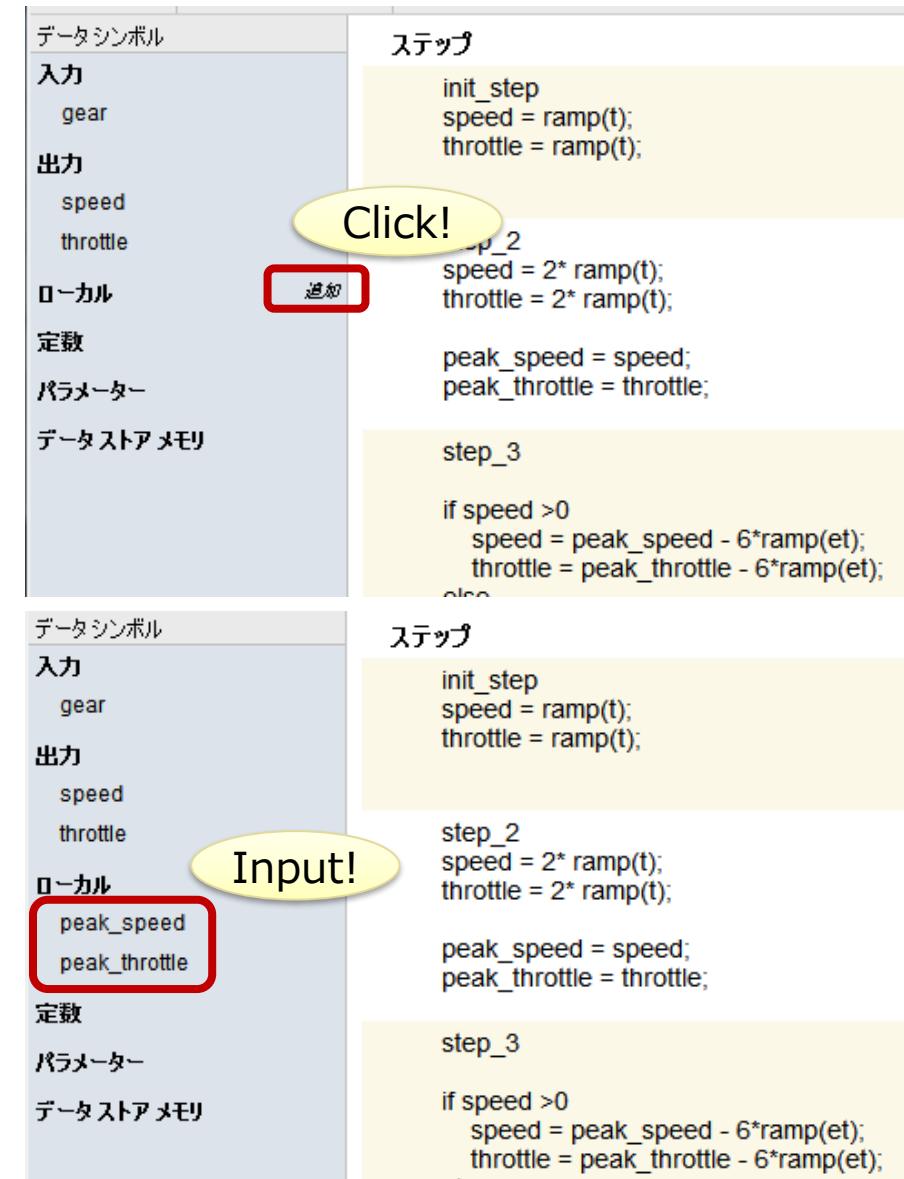
- [step_2] のステップ欄の右下にマウスをホバリングし、表示された [後にステップを追加] をクリックします。
- [step_3] の作業を右図のように定義します。
- [step_2] の遷移欄の右下にマウスをホバリングし、表示された [遷移の追加] をクリックします。
- [step_2] からの遷移条件に [gear == 3] を入力し、遷移先に [step_3] を指定します。



演習 2) シーケンシャルなテスト作成

3. ローカル変数の定義

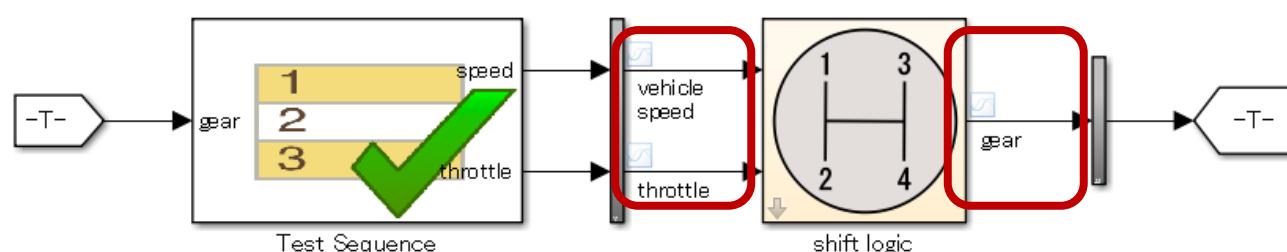
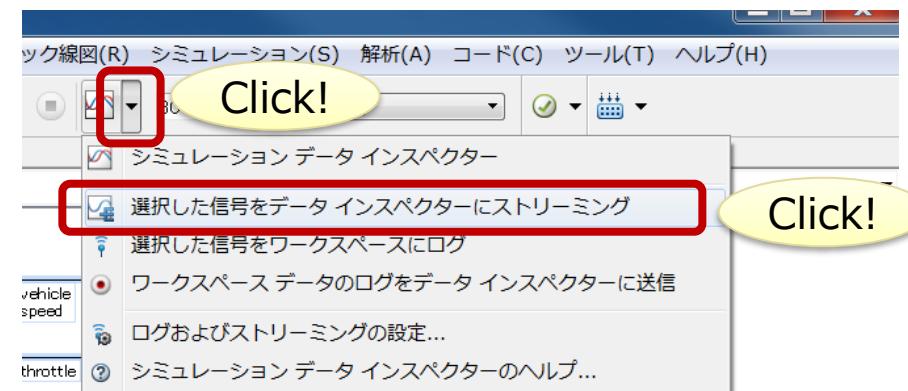
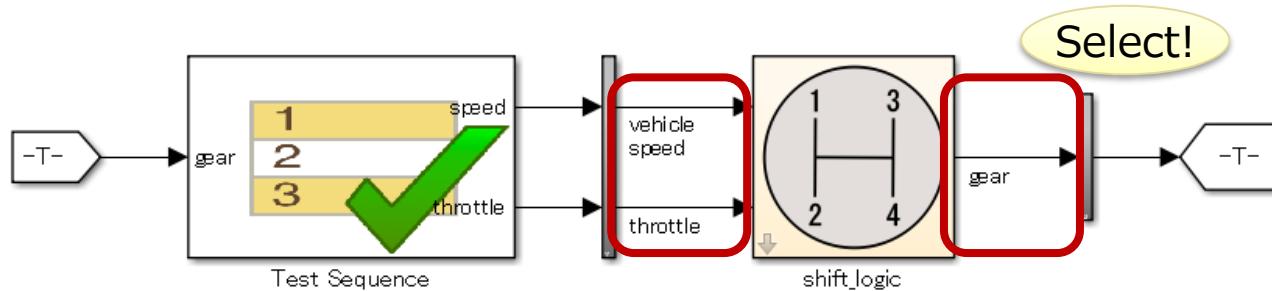
- テスト入力作成するために、[step_2] と [step_3] で使われている [peak_speed] や [peak_throttle] をこのテストシーケンスブロックのローカル変数として定義します。
- [データシンボル] の [ローカル] にマウスカーソルをホバリングします。表示された [追加] をクリックし、[peak_speed] と [peak_throttle] を入力します。



演習 2) シーケンシャルなテスト作成

4. ストリーミング信号を指定します。

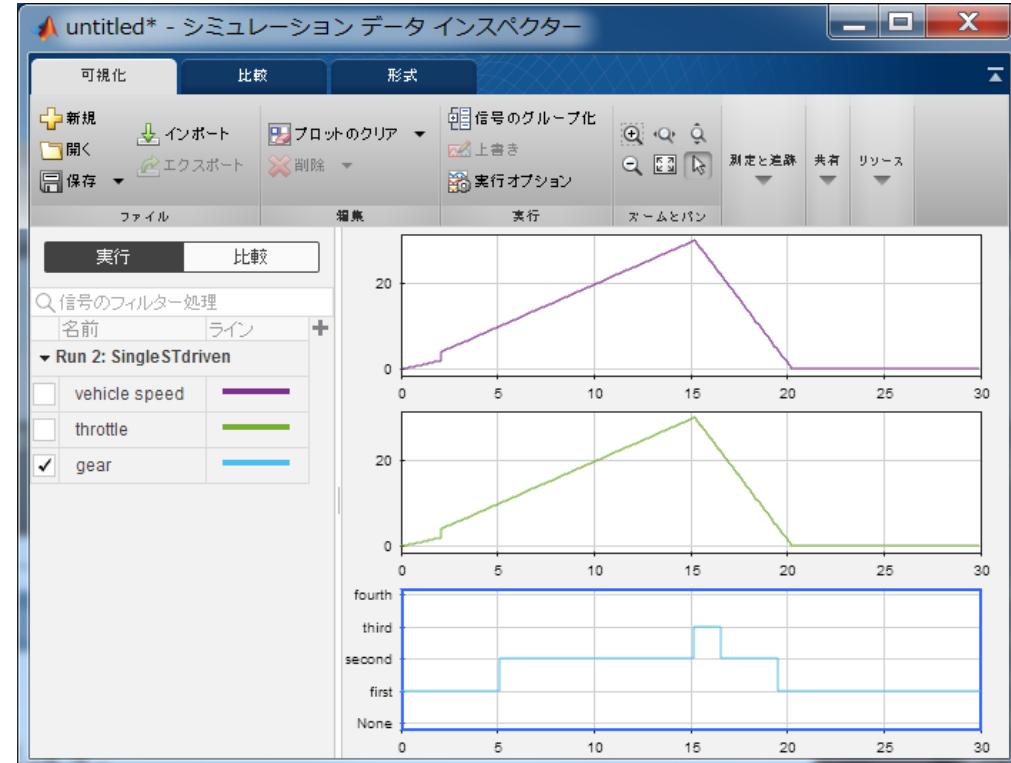
- [SingleSTdriven] のトップ階層に戻り、[vehicle speed]、[throttle]、[gear] の信号線をマウスカーソルで選択します。
- ツールバーにある
[シミュレーション データ インスペクター]  の
プルダウンメニューから
[選択した信号をデータ インスペクターにスト
リーミング] を選択します。
- 選択した信号にストリーミングのマーク  が表示されます。



演習 2) シーケンシャルなテスト作成

5. シミュレーションを実施し、結果を確認します。

- [実行] ボタン  をクリックし、シミュレーションを実施します。
- ハイライトされている [シミュレーションデータ インスペクター]  をクリックし、データインスペクターを開きます。
- ステップ4で設定した [vehicle speed]、[throttle]、[gear] のストリーミング信号が左側に表示されます。
- 右側のグラフに、上から順番に [vehicle speed]、[throttle]、[gear] 信号をアサインします。
- [vehicle speed] と [throttle] 信号が定義通りに、2秒に加速2倍になり、出力信号のギアが3速ギアになったタイミングで減速信号に切り替わったことが確認できます。

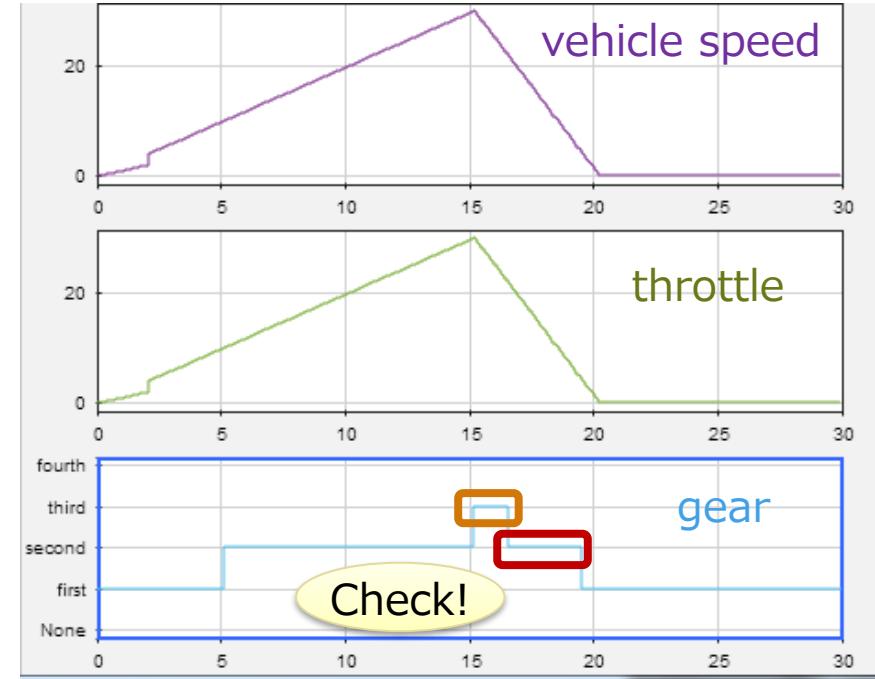


メリット④：複雑な
入力パターンも簡潔
に定義可能

演習 2) シーケンシャルなテスト作成

6. 診断項目を定義します – その1

- [減速時のギアが3速ギアになっているのは、必ず車速が16km/h以上の場合のみ] と [減速時のギアが、2速ギアになってから2秒以内に更にギアシフトすること] を診断項目として定義します。
- テストハーネス [SingleSTdriven] に戻り、[Test Sequence] をダブルクリックして開きます。
- [step_3] のステップ欄の右下にマウスをホバリングし、表示された [サブステップを追加] をクリックします。



演習 2) シーケンシャルなテスト作成

6. 診断項目を定義します – その 2

- 右図のように時相関数の ‘when’ と診断関数の ‘assert’ を使用し、[step_3_1] に [減速時のギアが3速ギアになっているのは、必ず車速が16km/h以上の場合のみ] を診断項目として定義します。
- [step_3_1] のステップ欄の右下にマウスをホバリングし、表示された [後にステップを追加] をクリックします。
- 右図のように時相関数の ‘when’ や ‘et’ と診断関数の ‘assert’ を使用し、[step_3_2] に [減速時のギアが、2速ギアになってから2秒以内に更にギアシフトすること] を診断項目として定義します。

**メリット⑤：診断項目
もテスト入力と一緒に
定義可能**

```
function step_3
if speed >0
    speed = peak_speed - 6*ramp(et);
    throttle = peak_throttle - 6*ramp(et);
else
    speed = 0;
    throttle =0;
end
```

step_3_1 when gear == 3
assert (speed >= 16, 'speed < 16 in 3rd gear');

Input!

step_3_1 when gear == 3
assert (speed >= 16, 'speed < 16 in 3rd gear');

Click!

後にステップを追加

サブステップを追加

step_3_1 when gear == 3
assert (speed >= 16, 'speed < 16 in 3rd gear');

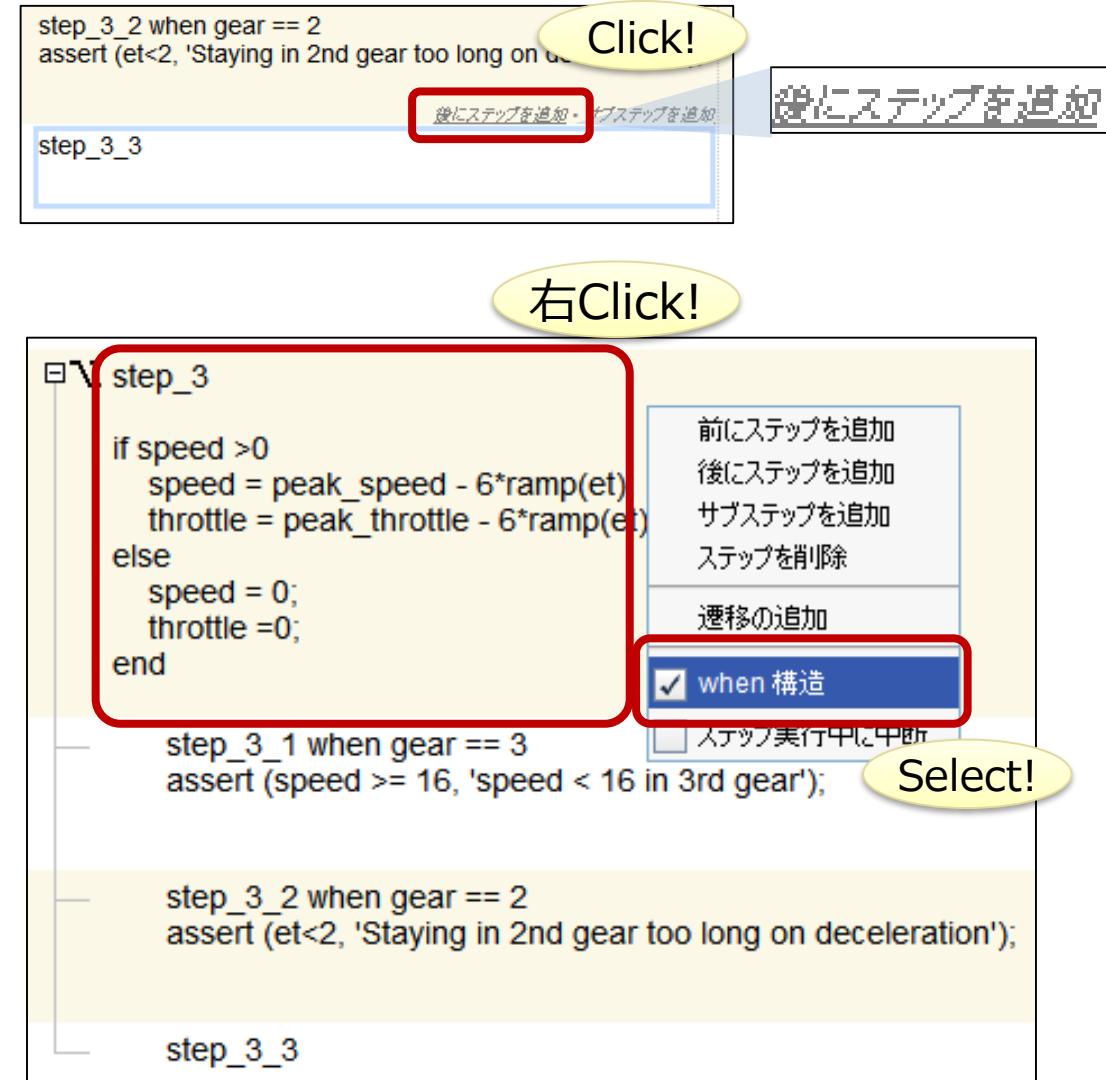
step_3_2 when gear == 2
assert (et<2, 'Staying in 2nd gear too long on deceleration');

Input!

演習 2) シーケンシャルなテスト作成

6. 診断項目を定義します – その3

- [step_3_2] の後に when 構造の締めステップとして空っぽの [step_3_3] を追加します。
- ‘when’ を有効させるのに、[step_3] を右クリック → [when構造] にチェックを入れます。
- しますと、[step_3] にwhen構造のアイコンが表示されます。

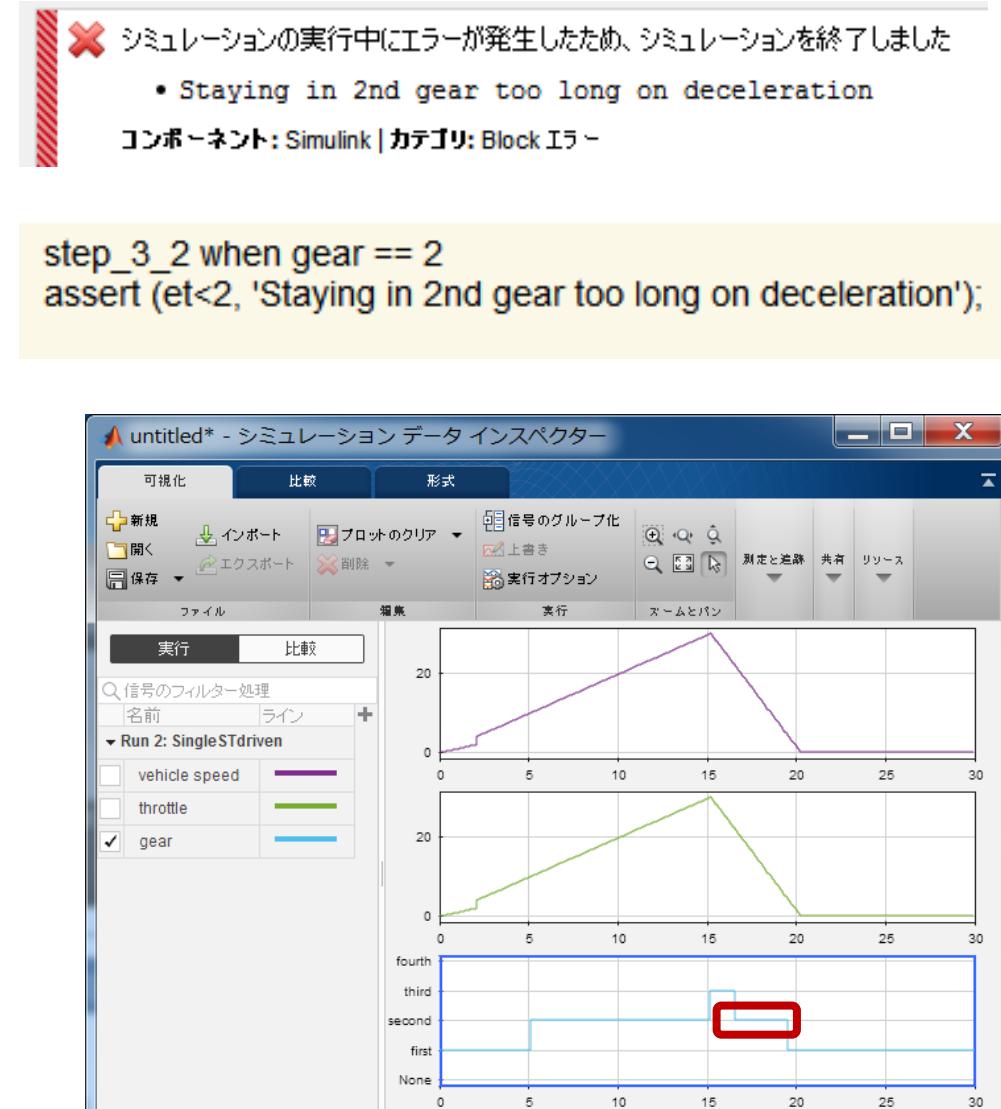


演習 2) シーケンシャルなテスト作成

7. シミュレーションを実施し、結果を確認します。

- [実行] ボタン  をクリックし、シミュレーションを実施します。
- 右図のようなエラーが発生します。
- [step_3_2] で定義されている [減速時のギアが、2速ギアになってから2秒以内に更にギアシフトすること] の検証項目を違反していることが分かります。
- (ステップ 5 のストリーミング波形でも、減速時のギアが2速ギアになってから2秒以上経過することが確認できます。)

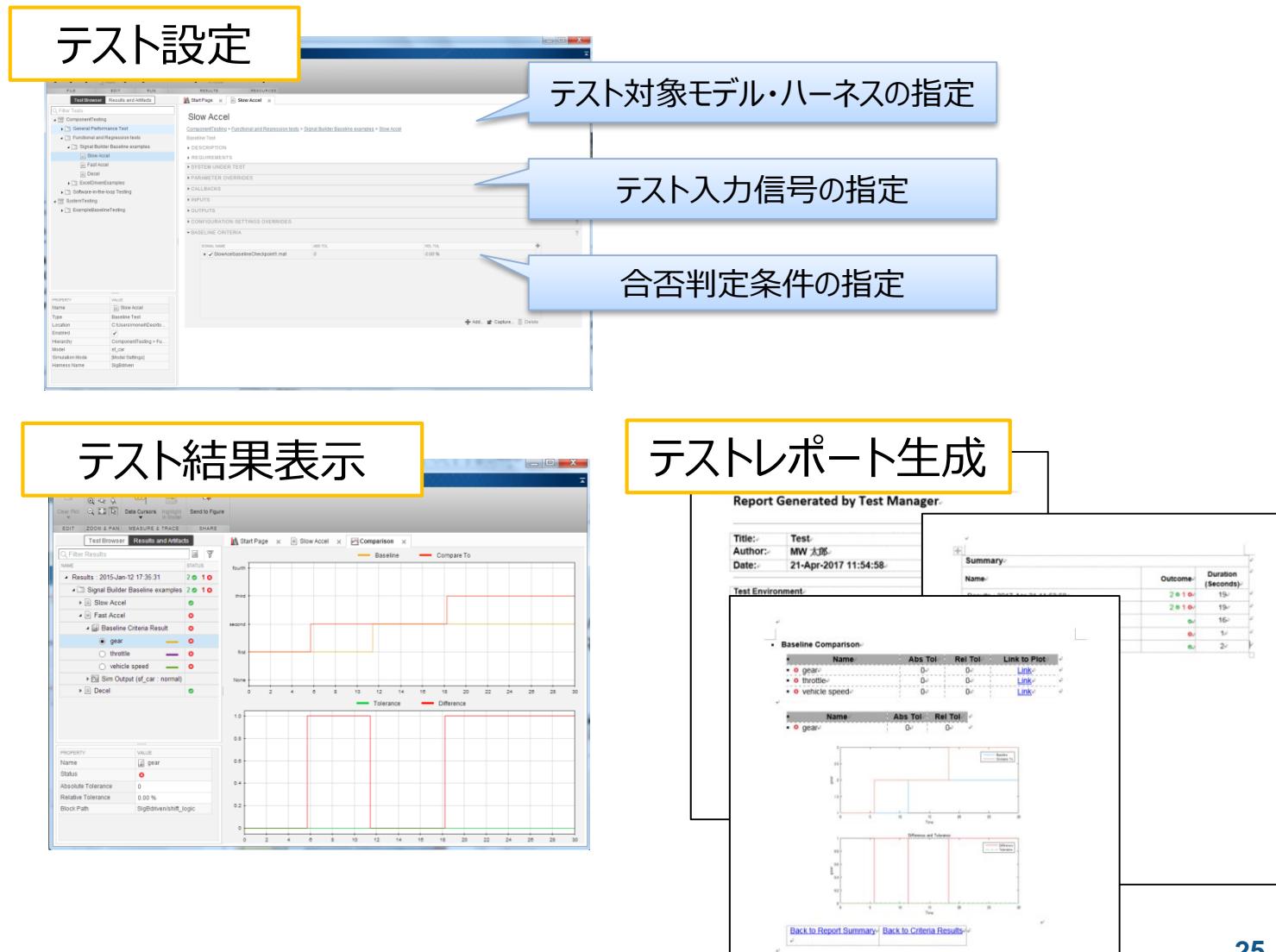
メリット⑥：診断項目に対するシミュレーション結果の目視チェックを省ける



テストマネージャー

複数テストを自動実行して合否レポートを作成可能

- 複数テストを統合管理
- バッチ処理による一括テスト実行
- テスト対象としてモデル全体
or テストハーネスを指定可能
- テスト結果レポート作成
- Excelからの信号読み込みに対応
- テスト時パラメータ上書き
- テスト用コールバック処理
- MIL/SIL/PILに対応



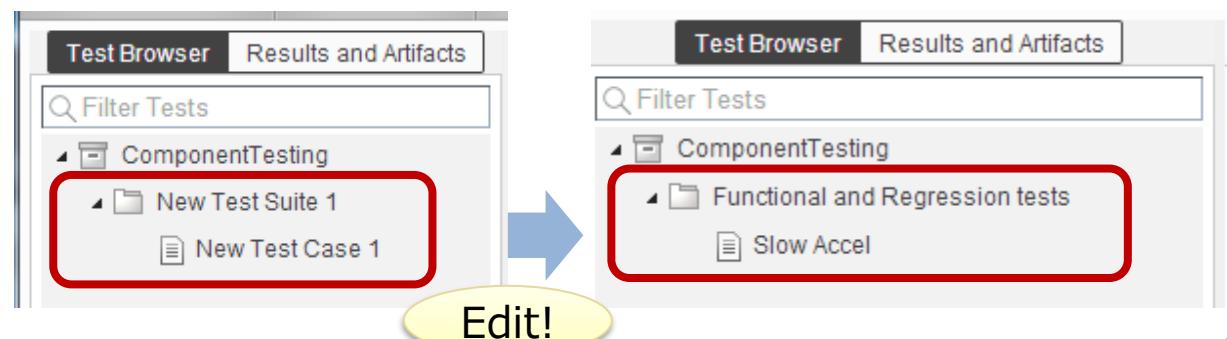
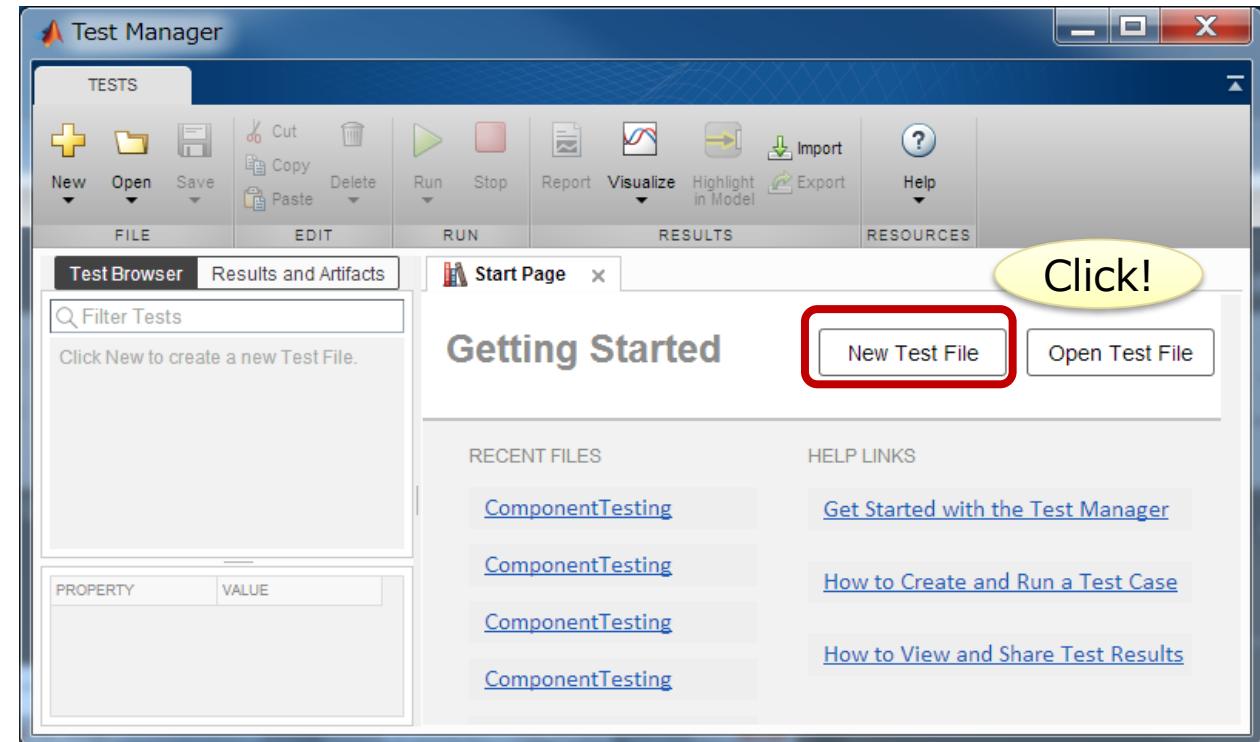
演習 3) 定型テストの自動化

1. 新規テストファイルを作成します。

- メインモデル [sf_car] → [解析] メニュー → [テスト マネージャー] をクリックし、テスト マネージャーの画面を開きます。
- [New Test File] → [ComponentTesting] のファイル名で新規テストファイルを保存。

2. テストスイートやテストケースの名前を変更します。

- [New Test Suite 1] → 右クリック → Rename → [Functional and Regression tests] を入力します。
- [New Test Case 1] → 右クリック → Rename → [Slow Accel] を入力します。



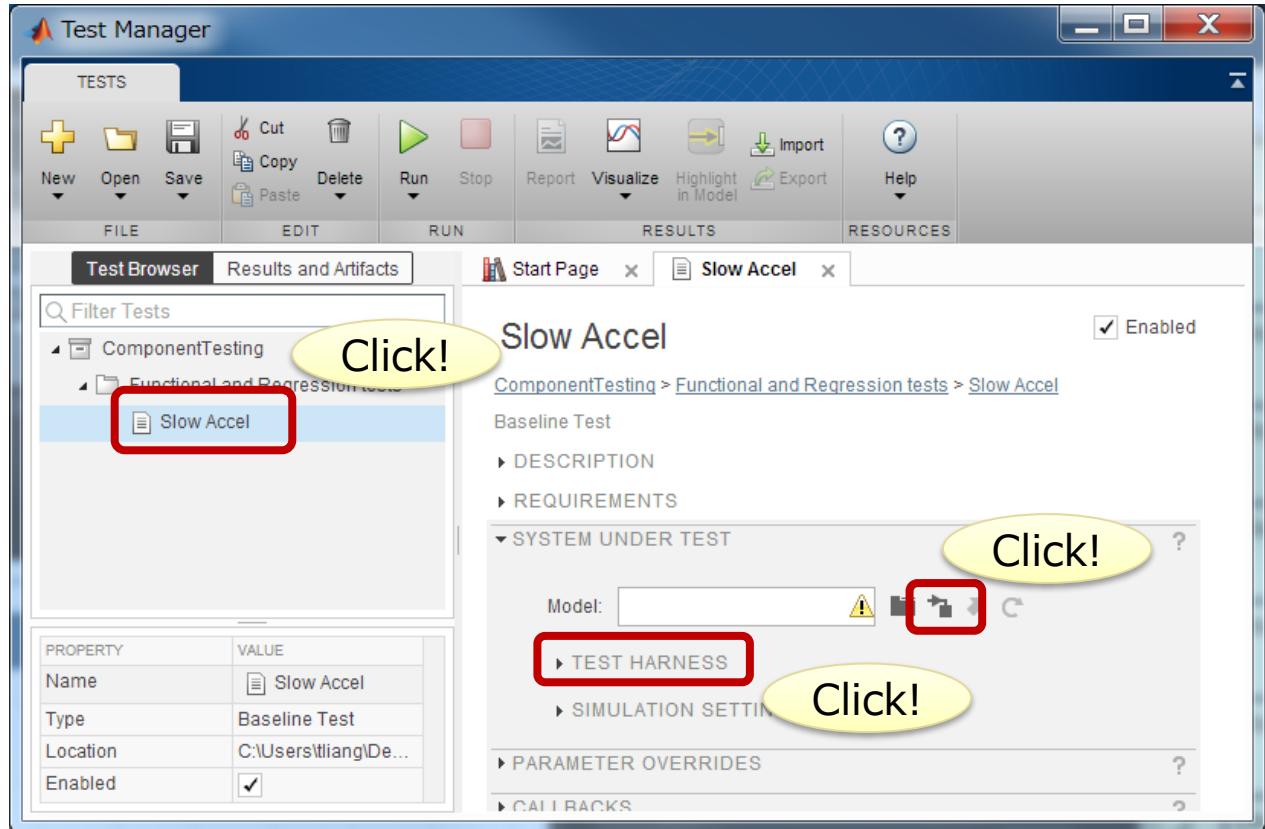
演習 3) 定型テストの自動化

3. テストケース [Slow Accel] の設定画面を開きます。

- テストマネージャーの左側にある [Slow Accel] をクリックします。
- テストマネージャーの右側に [Slow Accel] の設定画面が表示されます。

4. テスト対象を指定します。

- [SYSTEM UNDER Test] → Model
→ アイコンをクリックし、現在開かれているモデルを指定します。
- [TEST HARNESS] → リフレッシュアイコン → Harness のプルダウンリストからテストハーネスの [SigBdriven] を選びます。



演習 3) 定型テストの自動化

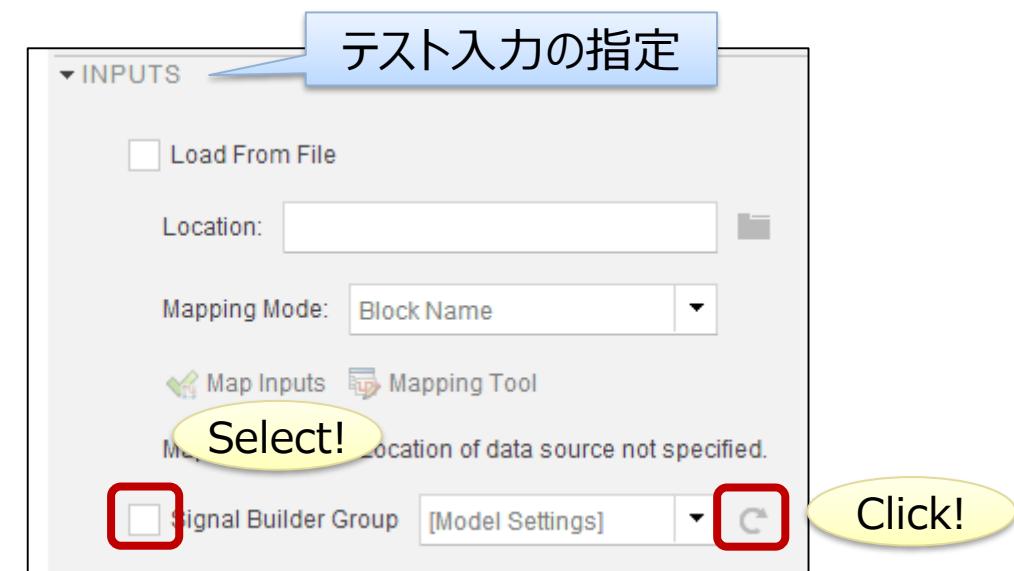
5. テスト対象を確認します。

- [Model] →  をクリックし、指定したモデルを開いて中身を確認できます。
- [Harness] →  をクリックし、指定したテストハーネスを開いて中身を確認できます。



6. テスト入力を設定します。

- [INPUTS] → [Signal Builder Group] にチェックを入れて、テストハーネス内の Signal Builder 信号を利用します。
- リフレッシュアイコン → テスト入力プルダウン  から [SlowAccelerate] をテスト用入力信号として指定します。



演習 3) 定型テストの自動化

7. 合否判定を定義します。

- [BASELINE CRITERIA] → [Add] → [SlowAcelbaselineCheckpoint1.mat] を開き、期待値ファイルとして指定します。
- 絶対許容誤差値 “ABS TOL” や相対許容誤差 “REL TOL” を [0] のままにします。

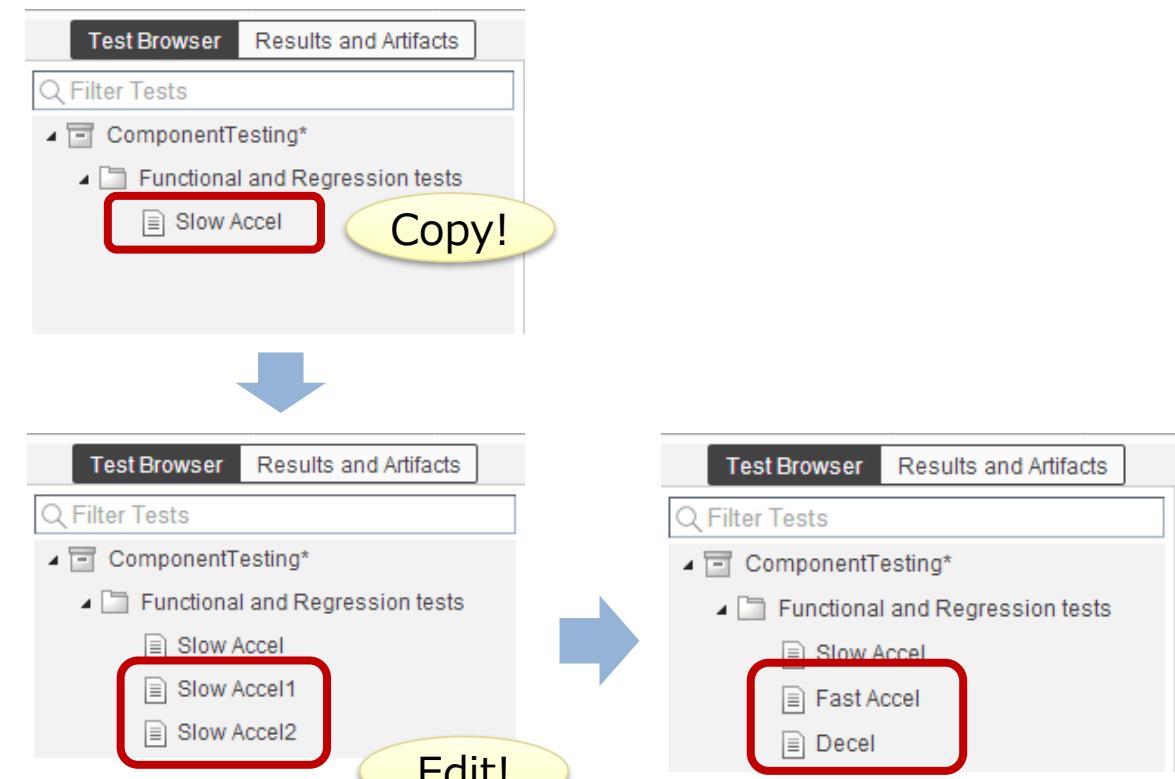


メリット⑦：テスト環境
(テスト対象・入力・期待値
など)を一括管理可能

演習 3) 定型テストの自動化

8. テストケースを複製します。

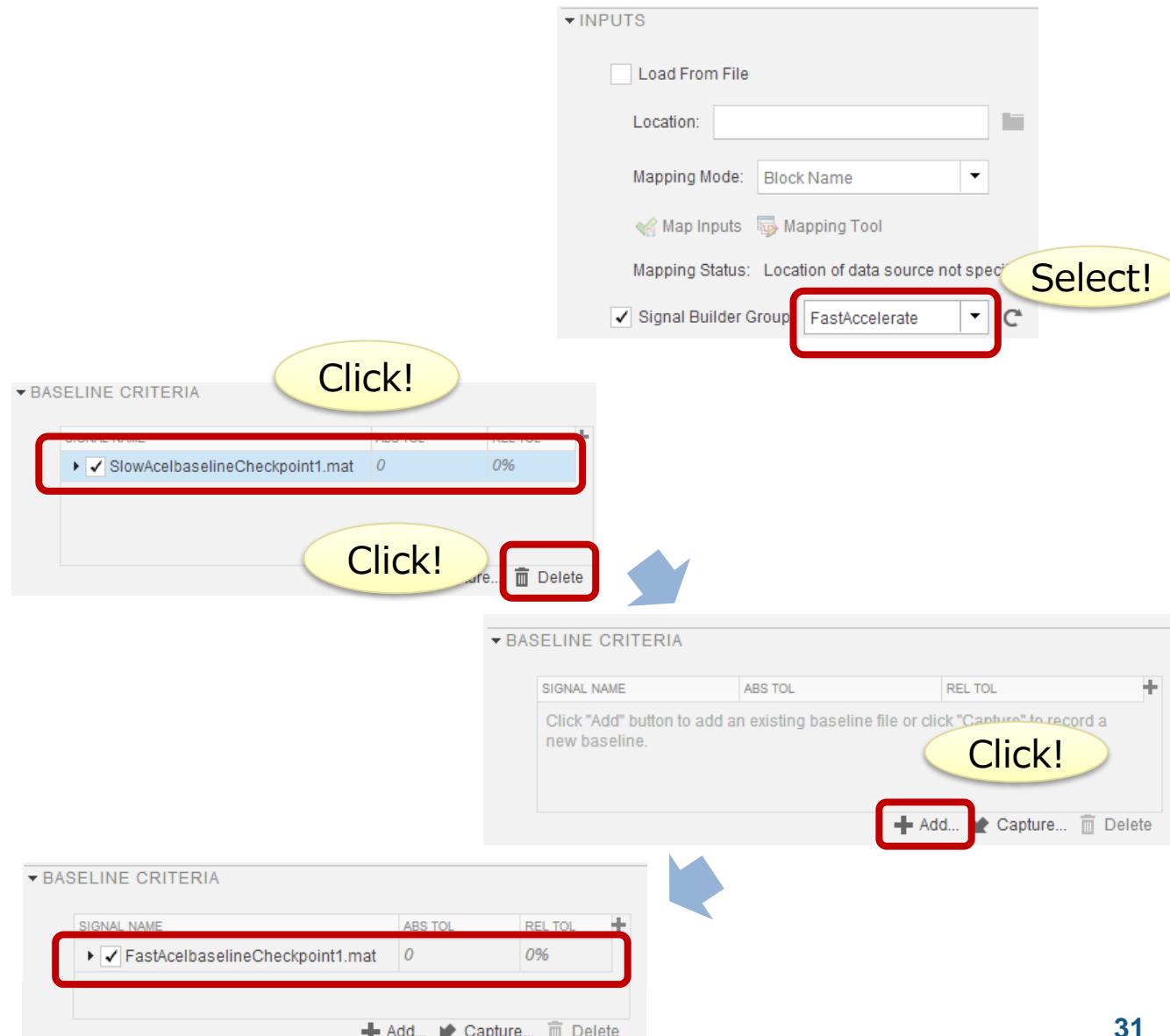
- [Slow Accel] → 右クリック → [Copy] → [Paste] x2 でテストケースの [Slow Accel] を2回複製します。
- [Slow Accel1] → 右クリック → [Rename] → [Fast Accel] に名前を変更します。
- [Slow Accel2] → 右クリック → [Rename] → [Decel] に名前を変更します。



演習 3) 定型テストの自動化

9. テストケース [Fast Accel] のテスト入力や合否判定条件を編集します。

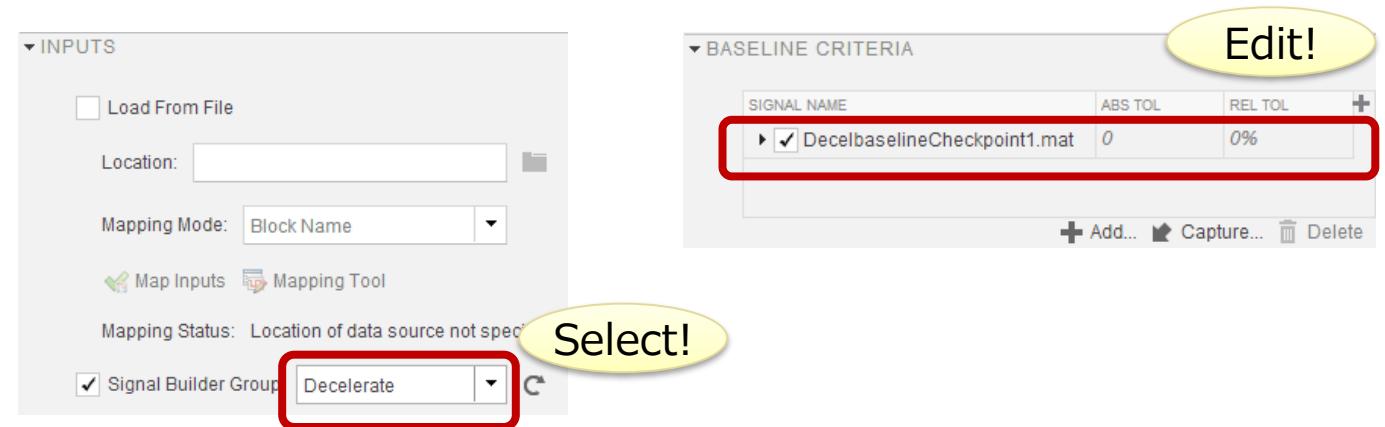
- テストマネージャーの左側にある [Fast Accel] をクリックします。
- [INPUTS] → [Signal Builder Group] → [FastAccelerate] をテスト用入力信号として指定します。
- [BASELINE CRITERIA] → [SlowAcelbaselineCheckpoint1.mat] をクリック → [Delete]。
- [BASELINE CRITERIA] → [Add] → [FastAcelbaselineCheckpoint1.mat] を開き、期待値ファイルとして指定します。



演習 3) 定型テストの自動化

10. テストケース [Decel] のテスト入力や合否判定条件を編集します。

- ステップ9と同じ手順で、テストケース [Decel] の設定画面を開きます。
- テスト入力に [Decelerate] を指定し、合否判定用の期待値ファイルに [DecelbaselineCheckpoint1.mat] を指定します。

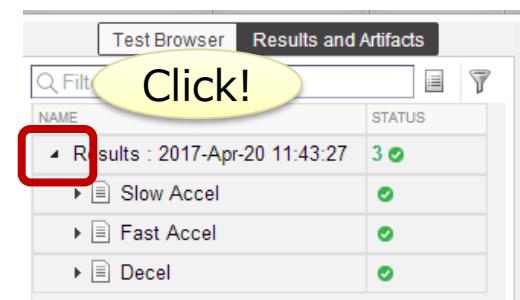
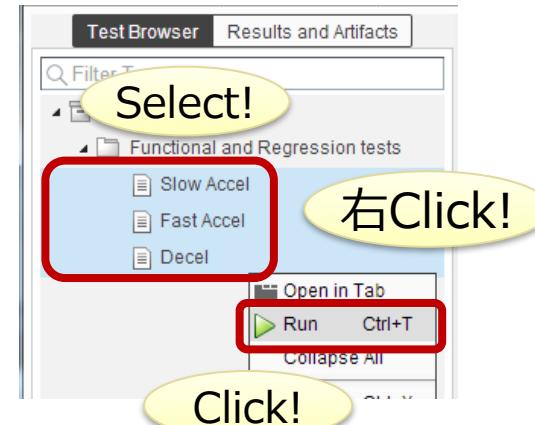


メリット⑧：テストケース
を簡単に流用可能

演習 3) 定型テストの自動化

11. テストケースを一括実行します。

- テストケース [Slow Accel]、[Fast Accel]、[Decel] を選んで → 右クリック → [Run] で選択されている 3 つのテストケースを一括実行します。
- [Results] でテスト結果のサマリー (3つとも Pass) を確認できます。
- [Results] を展開しますと、それぞれのテストケースの合否判定結果を確認できます。



NAME	STATUS
Results : 2017-Apr-20 11:43:27	3 ✅
Slow Accel	✅
Fast Accel	✅
Decel	✅

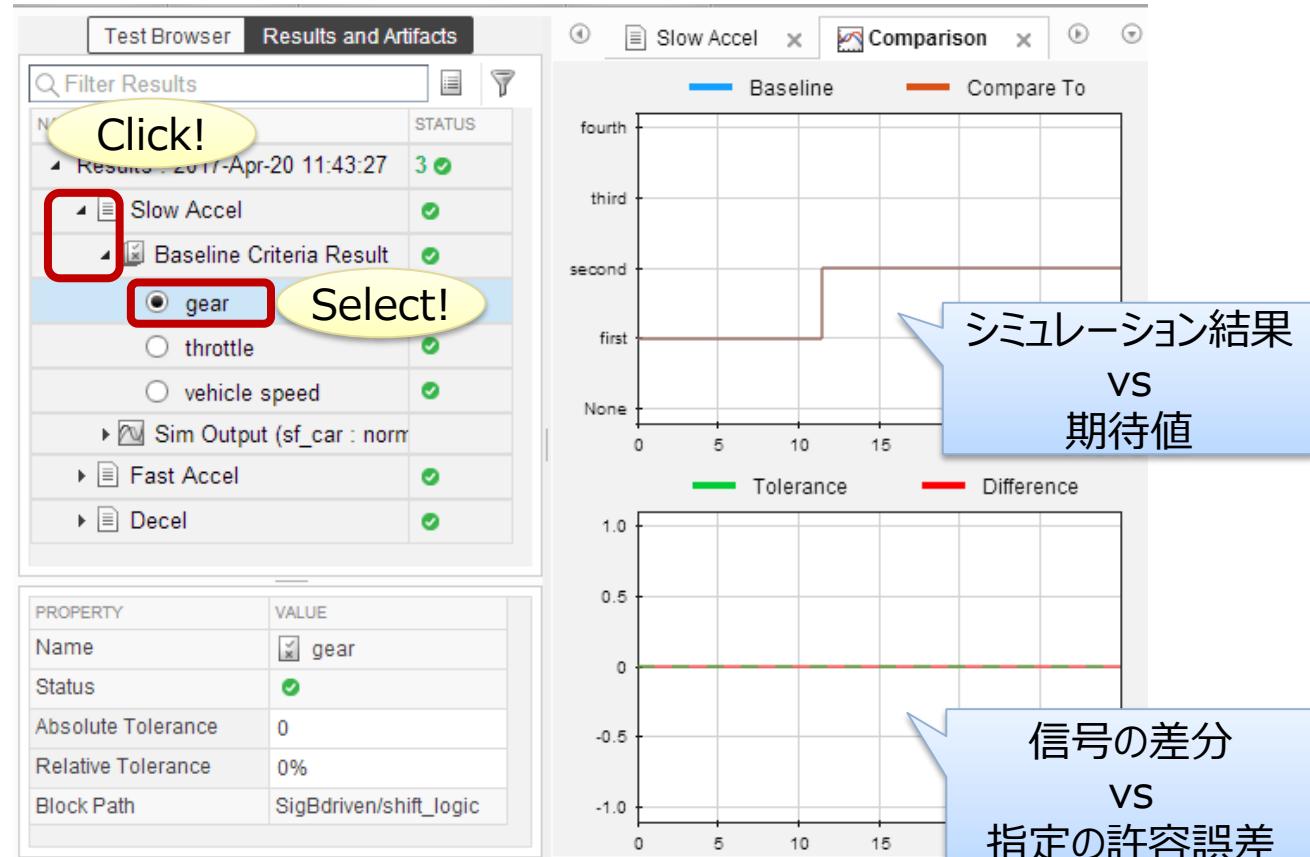
The screenshot shows the 'Results' table in the MATLAB Test Browser. It contains four rows: a summary row with '3 ✅' and three test case rows: 'Slow Accel', 'Fast Accel', and 'Decel', each with a green checkmark. A red box highlights the first row. A yellow callout labeled 'Click!' points to the first row.

メリット⑨：複数テストケース
の一元管理・一括実行可能

演習 3) 定型テストの自動化

12. テスト結果の詳細を確認します。

- 更に、テストケースを展開しますと、それぞれの詳細情報を確認できます。
- [Slow Accel] → 展開 → [Baseline Criteria Result] → 展開 → [gear]
- [gear] 信号のプロットが画面の右側に表示されます。
上：シミュレーション結果や期待値のプロット
下：シミュレーション結果と期待値の差や設定された許容誤差のプロット
- [gear] のシミュレーション結果が期待値と一致しているため、Pass判定されました。



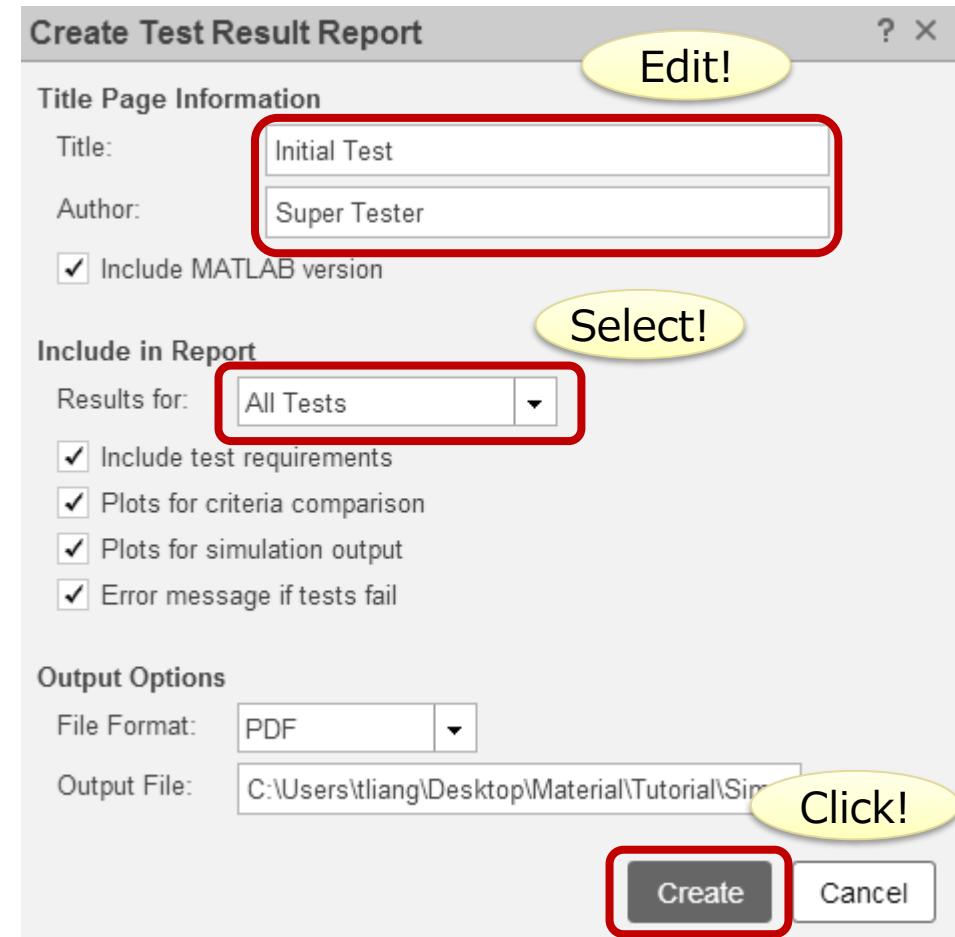
**メリット⑩：テストの自動実行
や自動判定でテスト工数を
低減可能**

演習 3) 定型テストの自動化

13. テストレポートを生成します。

- テスト結果 [Results : …] → 右クリック → [Create a report] → レポート生成画面が表示されます。
- レポートタイトル [Title] : Initial Test
- レポート作成者 [Author] : Super Tester
- レポートに含めたいテスト結果 [Include in Report] → [Results for] : All Tests
- [Create] を押して、レポートを生成します。

メリット⑪：レポートの自動生成で工数削減すること可能



コンテンツ

1. Simulink Test について
2. Simulink Test の基本操作の習得
 - テストハーネス作成
 - シーケンシャルなテスト作成
 - 定型テストの自動化
3. まとめ & 参考情報

必要・関連製品

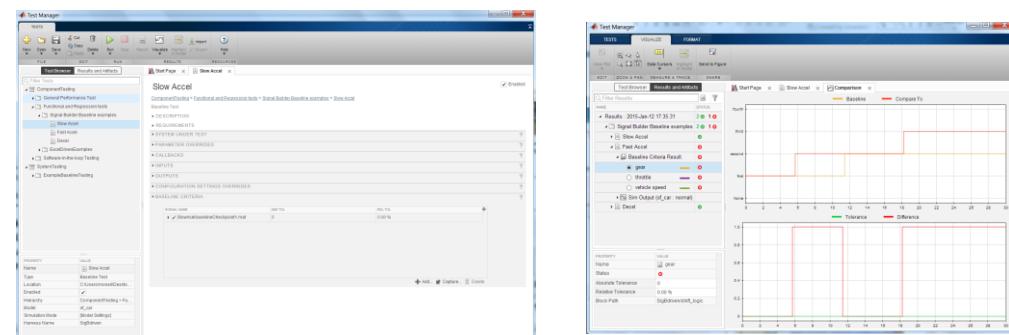
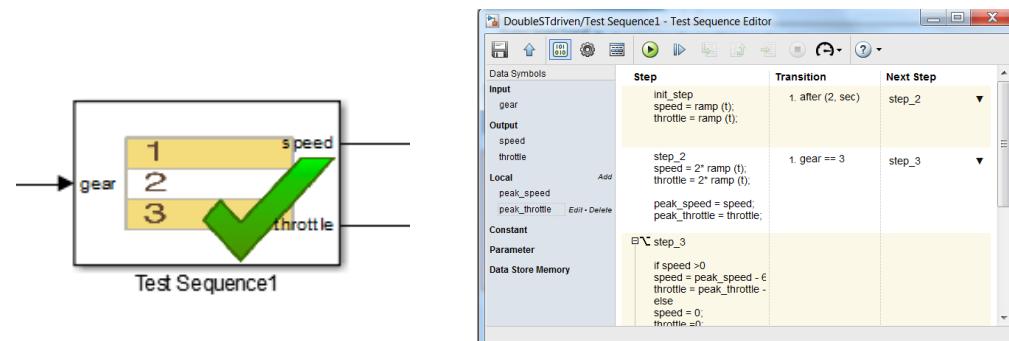
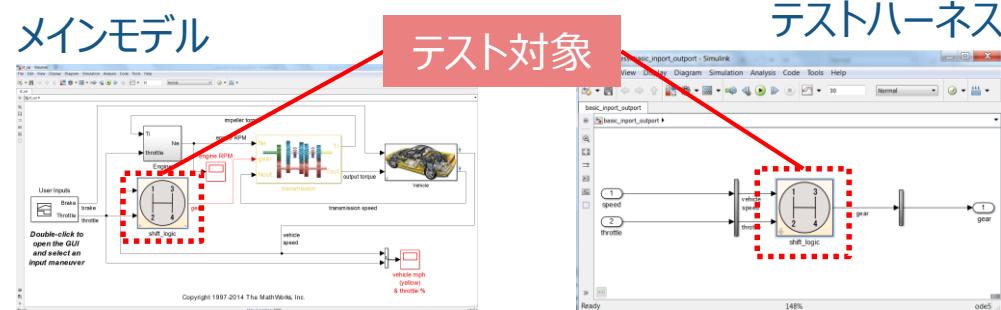
- 必要製品
 - MATLAB
 - Simulink
- 関連製品
 - Stateflow
 - Embedded Coder
 - Simulink Verification & Validation
 - Simulink Design Verifier
 - Simulink Report Generator

まとめ - MATLAB/Simulink Report Generatorの活用で……

- **統合・単体テストの一元化・管理の容易さ**
 - モデル全体・サブシステム単体・参照モデルに複数のテストハーネスを定義・実行できます

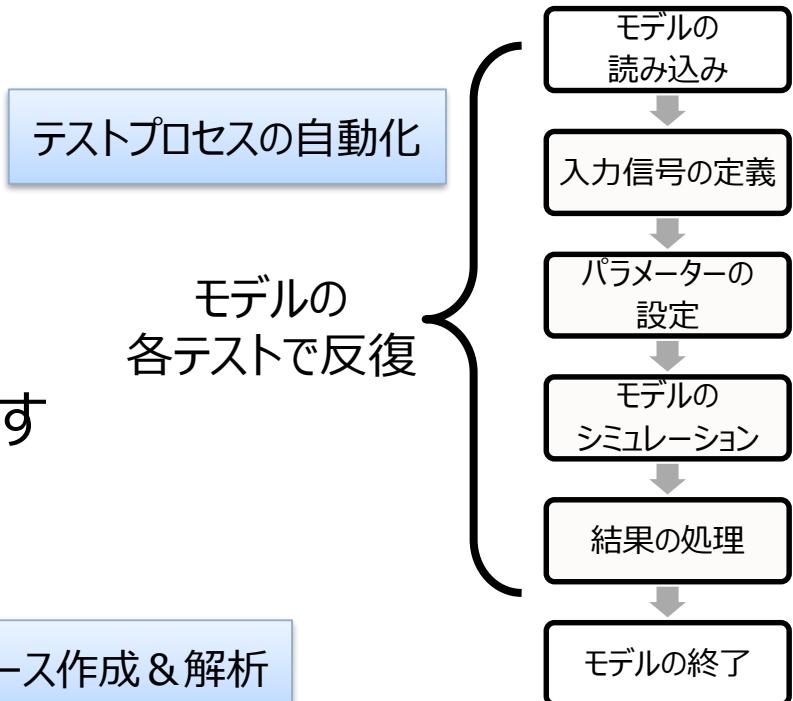
- **素早いタイミングチャート作り**
 - 複雑な入力パターンを状態遷移表で簡潔に表現できます

- **複数テスト・合否判定の効率化**
 - MIL/SIL/PILテストの自動化、複数テストの一括実行、レポート作成を行うことができます

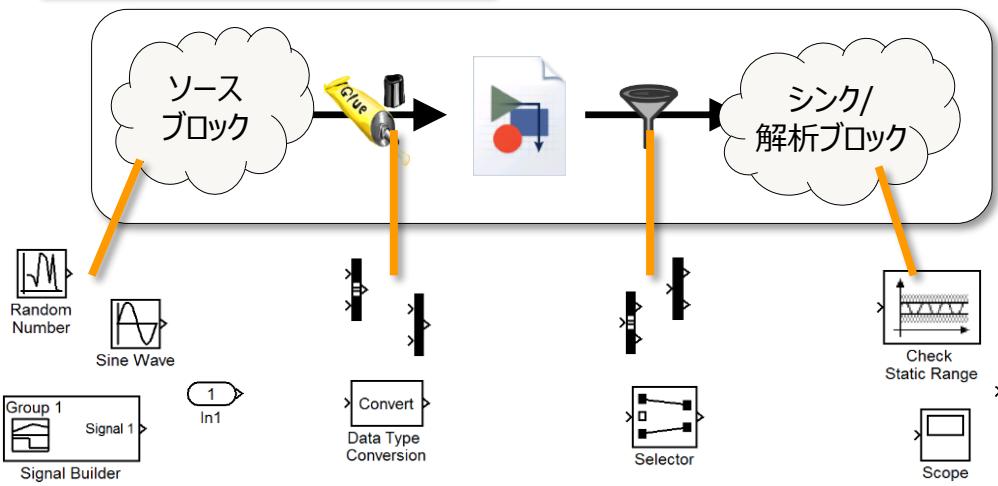


モデル検証に関するトレーニングコースのご紹介

- 『Simulink®モデルの検証と妥当性確認』
 - MATLAB&Simulinkおよび検証系オプションを活用したモデルの検証に関連する製品機能を体系的に学習できます
- このトレーニングで習得できること
 - Simulinkプロジェクトでのファイル管理
 - システム要求仕様のトレーサビリティ
 - テストケース作成方法の基本
 - テスト結果の可視化・結果の解析
 - テストプロセスの自動化
 - 検証結果のドキュメント化
 - カバレッジ計測 (Simulink Verification & Validation)
 - カバレッジ到達のための自動テスト入力生成 (Simulink Design Verifier)



テストケース作成 & 解析



付録

MathWorks の検証ツール

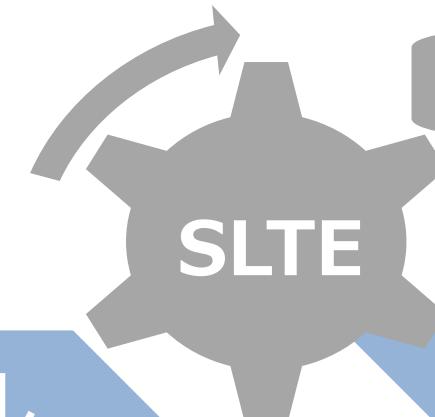
効果的かつ効率的なモデル検証を実現するための検証ツール群

Simulink Verification
and Validation™



a. モデル
設計

Simulink Design
Verifier™



b. モデル
作成

Simulink Test™

c. モデル
検証



Simulink Report
Generator™

効果的かつ効率的なモデル検証

Simulink Verification and Validation™



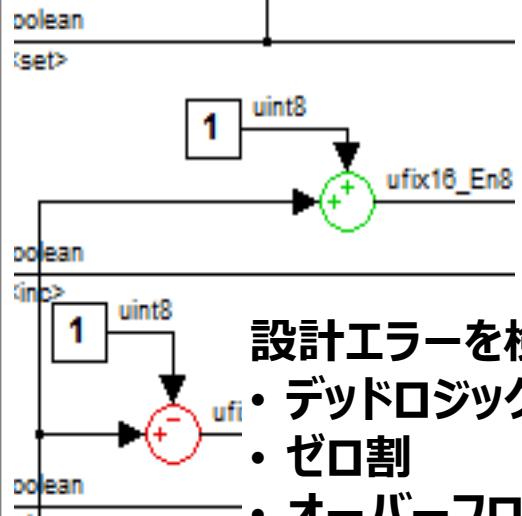
モデルカバレッジ測定	トレーサビリティ	モデルチェック
<p>入力データに対するモデルと S-Function コードのカバレッジを測定</p> <p>モデルカバレッジレポート</p> <ul style="list-style-type: none"> • Decision • Condition • MC/DC 	<p>相互リンクでモデルと仕様書間の行き来が可能</p> <p>モデルから仕様書へ</p> <p>仕様書からモデルへ (Word/Excel/DOORS)</p>	<p>ガイドラインに準拠したモデル記述になっていることを確認</p> <ul style="list-style-type: none"> • GUIのモデルチェック • 自動修正 • レポート生成 • カスタムチェックの追加 <p>警告 [Simulink I/O で厳密な型指定] は、以下のチャートでオフに設定されています。</p> <ul style="list-style-type: none"> • power_window_control_system_verif/act_control <p>推奨アクション 上記の Stateflow チャートで 'Simulink I/O で厳密な型指定' を選択してください。</p>

Simulink Design Verifier™



設計エラー検出

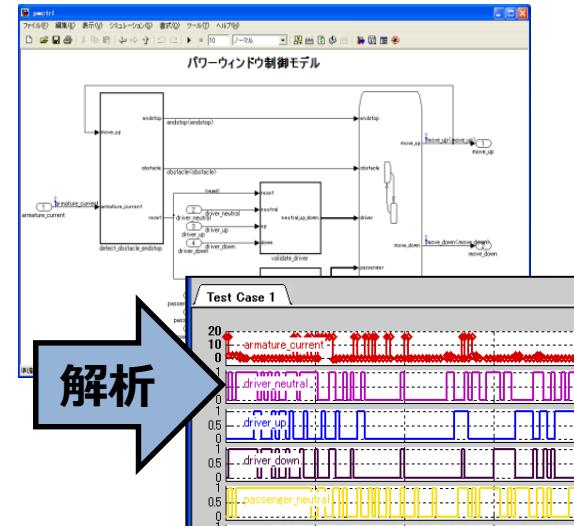
ゼロ割・オーバーフローなどの設計エラーが含まれていないかをチェック



テストケース自動生成

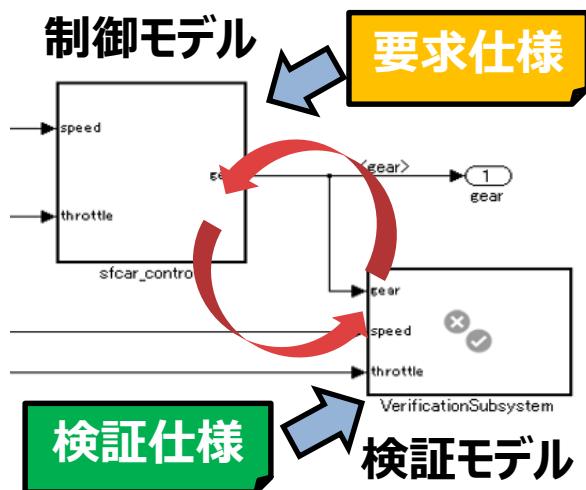
ロジックを網羅するテストケースを自動生成

制御モデル



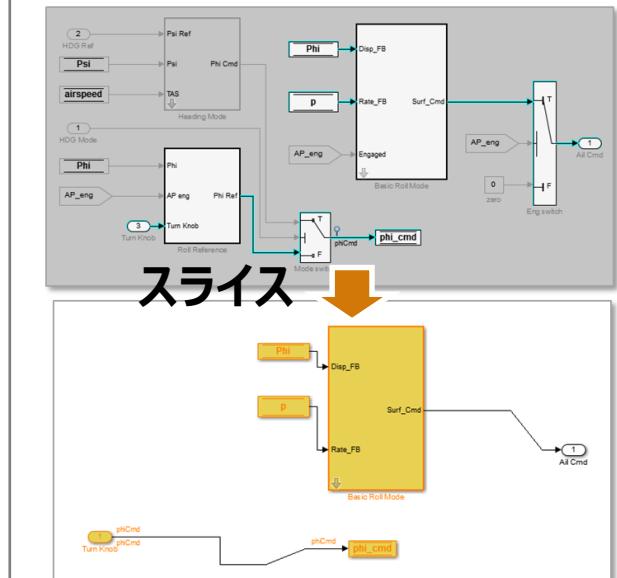
プロパティ証明 (形式検証)

取りうる入力範囲において、検証命題に矛盾がないことを証明

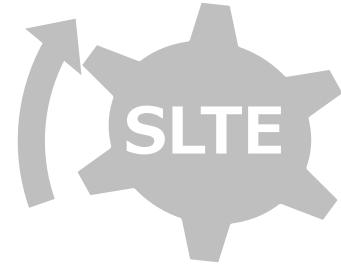


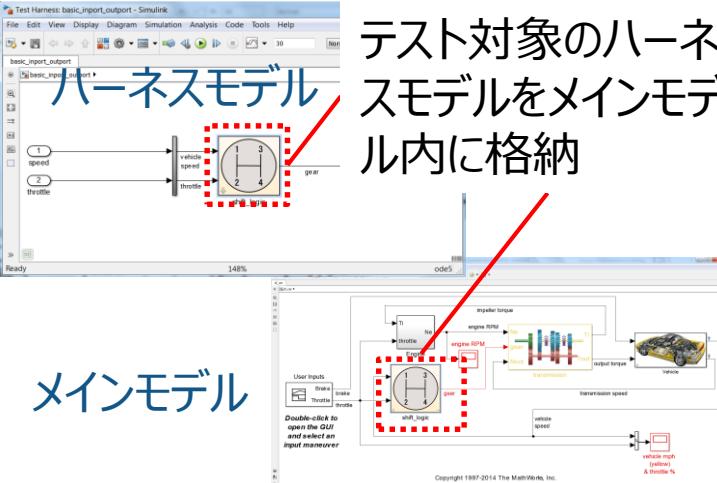
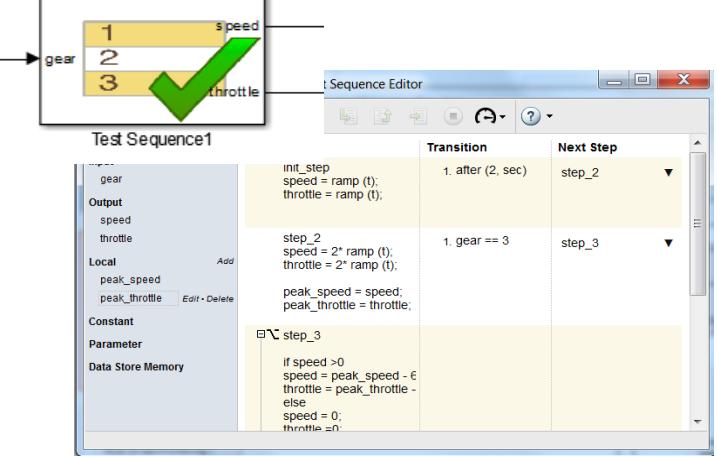
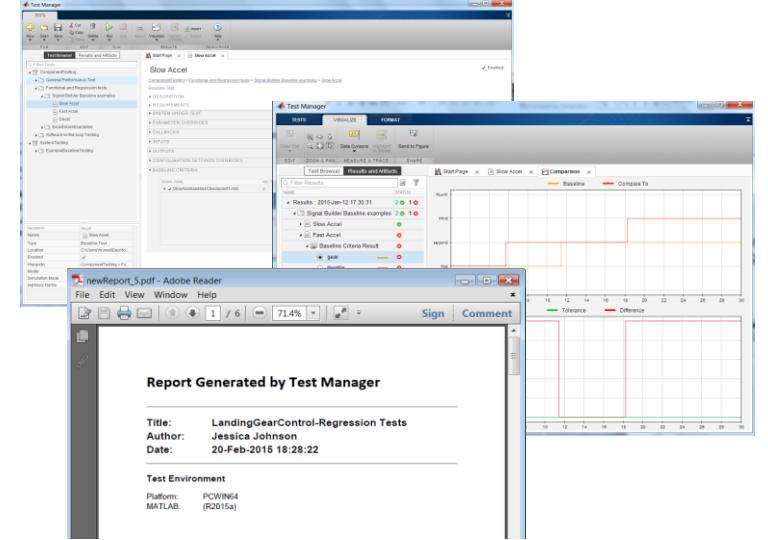
モデルスライサー

モデル内の信号依存性解析やその結果に基づくスライスモデルの作成が可能



Simulink Test™



テストハーネス	テストシーケンスブロック	テストマネージャー
モデル全体・サブシステム単体に対して複数テストハーネスを定義・実行	複雑な入力パターンを状態遷移表で簡潔に表現	MIL/SIL/PILテストの自動化、レポート作成、テスト管理
 <p>ハーネスモデル</p> <p>メインモデル</p>	 <pre> Test Sequence1 Step 1: gear == 1 Step 2: gear == 2 Step 3: gear == 3 </pre>	 <p>Report Generated by Test Manager</p> <p>Title: LandingGearControl-Regression Tests Author: Jessica Johnson Date: 20-Feb-2015 18:28:22</p> <p>Test Environment Platform: PCWIN64 MATLAB: R2015a</p>

Simulink Report Generator™



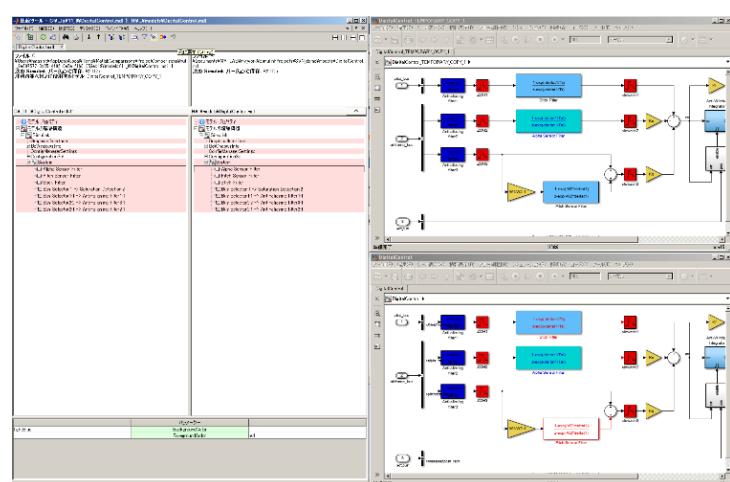
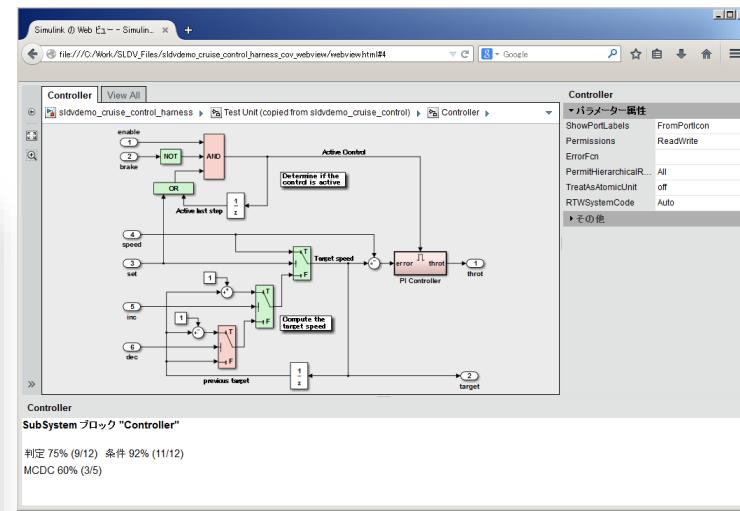
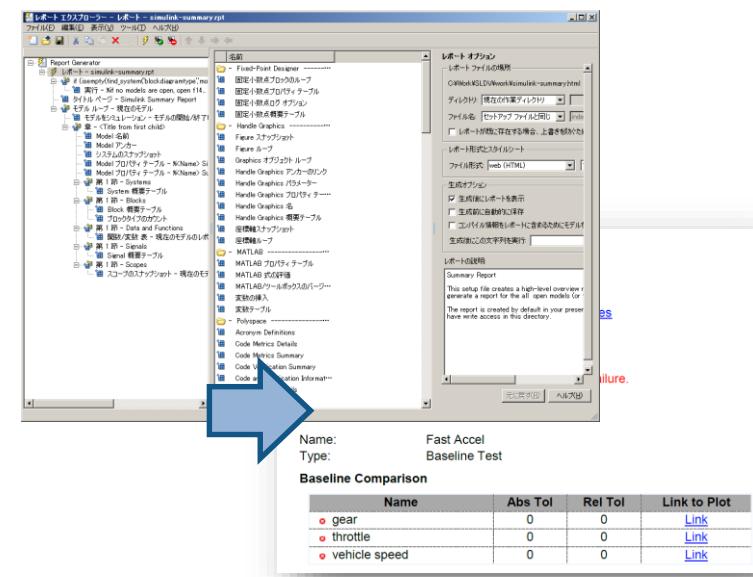
レポート生成

ユーザが定義したフォーマットに従ってレポートを自動生成

Web Export

Simulinkがない環境でも、モデルをブラウザで確認が可能(表示専用)

モデル差分比較・マージ





Accelerating the pace of engineering and science

© 2017 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.