特集論文

4 WD用電子制御装置のモデルベース開発

足立一幸*
村富義徳*
宇井信介*

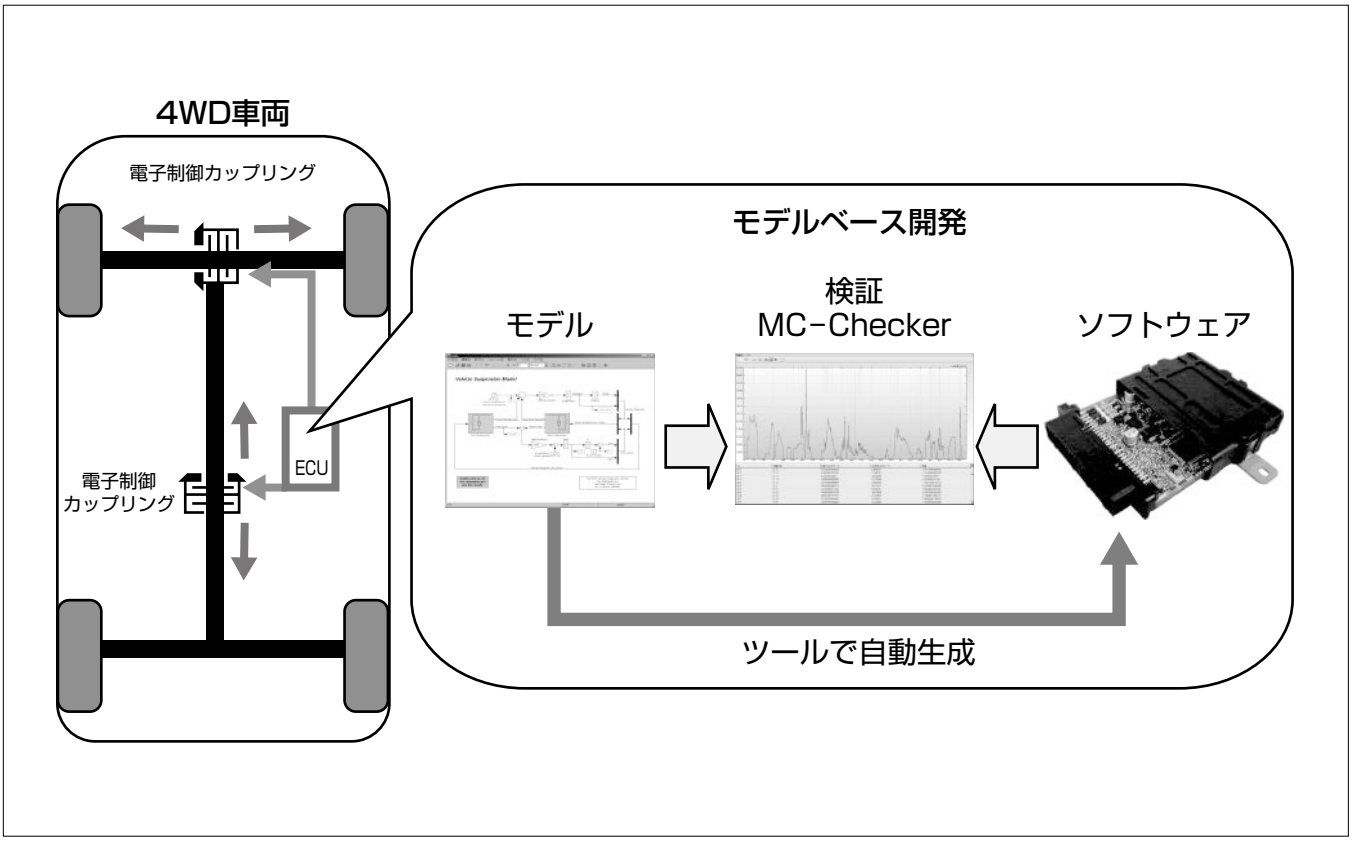
Model Based Development of Electronic Control Unit for 4WD

Kazuyuki Adachi, Yoshinori Muratomi, Shinsuke Ui

要　旨

モデルベース開発では，モデルと呼ばれる実行可能な制御フロー図で仕様を表現し，シミュレーションによって仕様の検証を行い，モデルからツールによってソフトウェアのソースコードを自動生成してソフトウェアを作成する環境が整いつつある。ツールによってソースコードを自動生成する環境は，従来のように要求仕様書を基に手作業でソフトウェアの作成を行う場合に比べて，大幅にソフトウェア作成工数を削減できるメリットがある。

しかし，増大するモデルの複雑さに比例して自動生成されるソースコードも増大し，その影響を受けてソフトウェアの検証工数は増大する一方である。今回，三菱電機はツールメーカーであるガイオ・テクノロジー㈱と共同でモデルとソフトウェアの一致性検証ツール“MC-Checker”を開発した。“MC-Checker”を使用することで，パソコン上でシミュレーションによって容易にモデルとソフトウェアの出力を比較したり，許容誤差を超えるソフトウェアの出力を検出できる環境が構築できた。実際に“MC-Checker”を三菱自動車工業㈱向け4 WD用電子制御装置（Elecrtronic Control Unit：ECU）のモデルベース開発に適用し，ソフトウェアの問題点を検出して効果をあげることができた。今後，モデル設計時に使用したモデルの検証データをソフトウェアの検証に再利用することで，効率化が図れると考える。



4WD用電子制御装置のモデルベース開発

三菱自動車工業㈱向け4WD用電子制御装置（ECU）は前輪左右の駆動力配分と前後輪の駆動力配分を行うための2つの電子制御カップリングを統合制御している。その電子制御装置の組み込みソフトウェア開発には，モデルベース開発を採用している。ガイオ・テクノロジー㈱と共同で開発したモデルとソフトウェアの一致性検証ツール“MC-Checker”を検証プロセスに適用して，モデルベース開発の課題であった検証プロセスの効率化を図った。

*姫路製作所

1. まえがき

近年，環境問題に対する自動車の燃費規制や排ガス規制の強化，故障診断の高機能化などの要因によって，ECUの組み込みソフトウェア（以下“ソフトウェア”という。）は複雑化・大規模化し，その開発工数は増大する方向にある。この肥大化するソフトウェアの開発工数に対して，開発効率向上の目的で，モデルベース開発という開発手法を採用するカーメーカーが増えつつある。

モデルベース開発では，制御仕様をモデルと呼ばれる実行可能な制御フロー図で表現し，実車などでシミュレーションを実行して制御仕様の検証を行い，モデルからツールによって自動でソースコードを生成し，ソフトウェアを作成する環境が整いつつある。モデルの開発ツールとしては，MATLAB/Simulink(注1)がよく知られている。ツールによってソースコードを自動生成する環境は，従来のように要求仕様書を基に手作業でソフトウェアの作成を行う場合と比べて，ソフトウェアの作成工数を大幅に短縮できるメリットがある。しかし，一方，ツールで自動生成したソースコードの信頼性を確認するための検証プロセスの工数は増大しつつあり，この検証プロセスの効率化をいかに進めていくかということが課題となっている。

本稿では，当社で取り組んだモデルベース開発の検証プロセス改善について，三菱自動車工業㈱向け4WD用ECUのモデルベース開発に適用した事例を通して述べる。

（注1） MATLAB，及びSimulinkは，The MathWorks，Inc.の登録商標である。

2. モデルベース開発の検証環境構築

2.1 検証プロセスの課題

モデルベース開発の検証プロセスでは，以下の内容についてソフトウェアの検証を行う。

⑴ 制御仕様が正しく実現できているか

車両制御が記述されるモデルは，記述内容や規模など多種多彩であり，ツールがあらゆるモデルに対して誤りのないソースコードを生成できるかを事前に確認することは一般的に難しい。そのため，モデルからツールでソースコードを自動生成するたびに，生成されたソースコードに誤りがないか確認する必要がある。

⑵ 固定小数点の使用による演算精度が十分であるか

当社のソフトウェアでは，固定小数点を1ビットあたりで表現する物理量とオフセットを設定した符号なし整数で表現する方法を採用している。ある一つの物理量を表現するのに，モデルでは倍精度浮動小数点を使用するため，メモリを8バイト使用するが，固定小数点を使用した場合のメモリ使用量は2バイト（4WD用ECUの場合）で済む。物理量を固定小数点で表現すれば，ソフトウェアのメモリ使用量が削減でき，ソフトウェアをメモリ容量の少ない安価なマイコンにも実装しやすくなる。しかし，一方で物理量を固定小数点で表現することによって，ソフトウェアの演算精度を落とすことにつながる。そのため，ソフトウェアの制御仕様で要求される演算精度(以下“許容誤差”という。)に注意しながら，固定小数点を部分的に使用しており，その影響によって演算精度が許容誤差を逸脱していないか確認する必要がある。

また前述のとおり，モデルの規模が増大することに比例して検証プロセスの工数は増加する傾向にあり，検証プロセスの効率化をどのようにして進めるかということもモデルベース開発の課題となっていた。

2.2 “MC-Checker”の開発

モデルベース開発の検証プロセス効率化の取組みとして，当社では2007年からツールメーカーであるガイオ・テクノロジー㈱と共同で仕様設計を行い，モデルとソフトウェアの一致性検証ツール“MC-Checker”の開発を行ってきた。“MC-Checker”は，**図1**に示すように大きく分けて以下の2つの機能を持つ。“MC-Checker”を使用することで，検証がパソコン上で自動実行できるようになり，モデルベース開発の検証プロセスの大幅な工数削減が見込めるようになった。

⑴ モデルとソフトウェアの出力を比較する機能

パソコン上で同一入力データをモデルとソフトウェアに与えて，それぞれシミュレーションを行い，モデルとソフトウェアの出力をグラフ表示して比較する機能である。

モデルはパソコン上でシミュレーションできるが，ソフトウェアは，本来ECUに搭載しているマイコン上で動作するものであり，パソコン上では動作しない。MC-Checkerでは，マイコンと同一の動作をパソコン上でシミュレーションできるマイコンエミュレータ上でソフトウェアを動作させることによって，ソフトウェアをECUに実装する形式のまま変更することなく，シミュレーションできるように構成している。

⑵ 許容誤差を超えるソフトウェアの出力を検出する機能

⑴で実現した機能によってモデルとソフトウェアの出力を比較したグラフ上で，ソフトウェアの出力がモデルの出

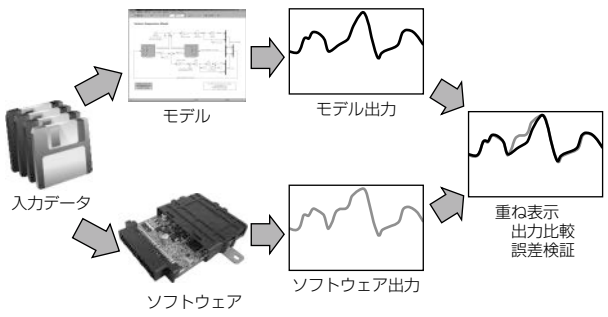


図1．MC-Checkerの機能

力を基準として，ユーザーが設定した許容誤差の範囲を超える箇所を検出し，マーカ表示する機能である。この機能によって，ソフトウェアの出力が許容誤差内にあるかどうかを視覚的に一目で確認できる。

2.3　4WD用ECUのモデルベース開発への適用

今回開発した“MC-Checker”を三菱自動車工業㈱向け4WD用ECUのモデルベース開発に適用した事例について述べる。

S-AWC(Super All Wheel Control)(注2)と呼ばれるこの4WD制御システムは，**図2**に示すように，S-AWC用ECUと前後輪の駆動力配分を行う電子制御4WD，前輪左右の駆動力配分を行うアクティブフロントデフ(AFD)から構成される。S-AWC用ECUは，AFD用のフロント電子制御カップリング(F-ECC)と電子制御4WD用のセンター電子制御カップリング(C-ECC)を統合制御する。S-AWC用ECUは，各種センサや他のECUからCAN(Controller Area Network)通信によって操舵(そうだ)角，車輪速，前後加速度などの車両の運転状況に関する情報を入手し，運転状況に応じて電子制御カップリングの係合トルクを制御し，前輪左右の駆動力配分と前後輪の駆動力配分を行うことで，車両の高い運動性能を実現している。

図3は，“MC-Checker”を使用し，実車両でS-AWC用ECUで取得した各種センサなどのデータを，ある開発段階でのモデルとソフトウェアに入力して，F-ECCの出力比較を実施した結果である。一部でモデルとソフトウェアでF-ECCの出力が大幅に異なっており，ソフトウェアに問題があることが分かった。

原因を解析したところ，4WD制御の状態遷移を表す状態変数について，モデルとソフトウェアで扱いに差があることが分かった。この状態変数は，車両のオーバーステア，アンダーステア状態を識別するために用いている。オーバーステアとは，車両が旋回するときに，後輪が旋回時の円の外側に滑り出して，ハンドルをきった角度よりもさらに旋回時の円の内側に小回りしてしまう状態をいう。アンダーステアとは，その逆の状態を示し，車両が旋回するときに，前輪が旋回時の円の外側に滑り出して，ハンドルをきった角度よりもさらに旋回時の円の外側に大回りしてしまう状態をいう。問題となった状態変数は，モデルでは**表1**のように値の割り当てがされていた。

モデルでは，この状態変数が浮動小数点で表現されていたため，すべての車両状態が正しく識別できていた。しかし，ソフトウェアは，この状態変数をオフセット0(マイナス値が表現できない設定)の符号なし固定小数点で表現していた。そのため，ソフトウェアでは，この状態変数が，値“－1”となるときのオーバーステアの車両状態が正しく表現できず，モデルとソフトウェアの出力の差の原因となっていた。

この状態変数の扱いを修正した後，再度“MC-Checker”

(注2)　S-AWCは，三菱自動車工業㈱の登録商標である。

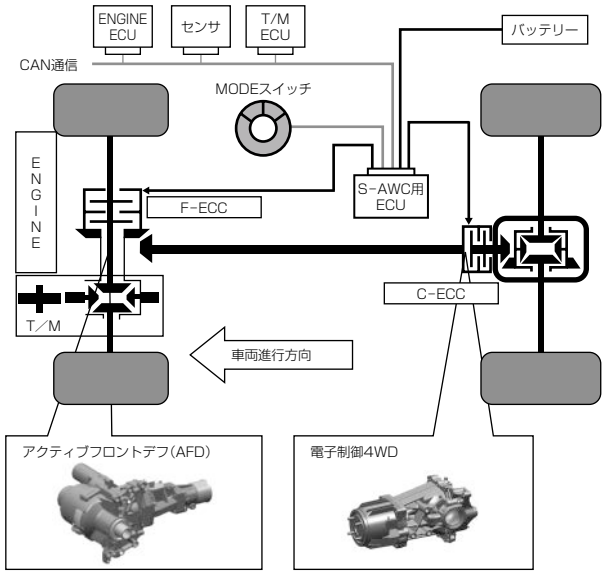


図2．4WDトランスミッション

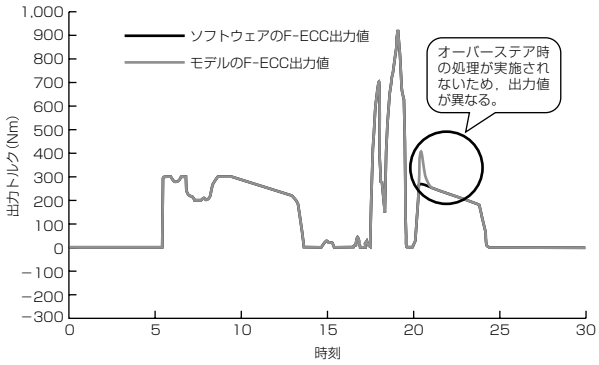


図3．F-ECCの出力比較(修正前)

表1．車両の状態遷移を示す変数

車両状態	状態遷移を示す変数の値	
	モデル	ソフトウェア
アンダーステア	1	1
直進状態	0	0
オーバーステア	−1	—

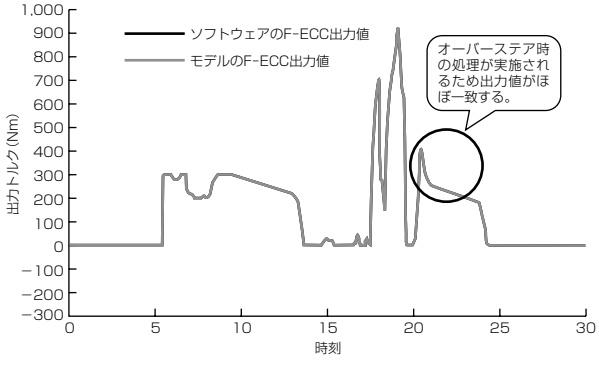


図4．F-ECCの出力比較(修正後)

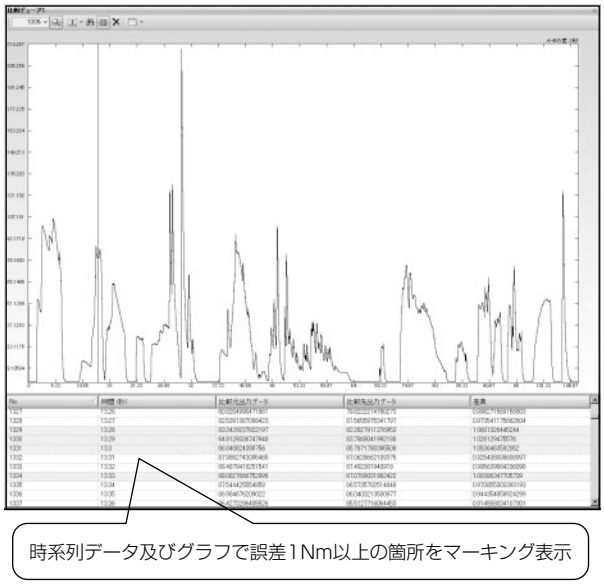


図5．許容誤差を超えるC-ECCの出力検出(従来)

を使用し，モデルとソフトウェアのF-ECCの出力比較を実施した結果を**図4**に示す。問題となっていた状態変数でオーバーステアの扱いが修正されたことによって，F-ECCの出力がほぼ一致し，ソフトウェアの動作に問題がないことが確認できた。

このように，“MC-Checker”を使用することで，モデルとソフトウェアの動作比較が容易に実施できる。

2.4　ソフトウェアの演算精度向上

“MC-Checker”を使用して，許容誤差を超えるソフトウェアの出力の検出を行った。**図5**に示すように，ソフトウェアでC-ECCの出力が許容誤差1Nmを超える誤差を検出できた。ソフトウェアの入力から出力に至るまでの演算制御の結果を順を追って確認したところ，固定小数点マップの演算処理でモデルとソフトウェアの出力差が大きくなることが分かった。

固定小数点マップとは，入力と出力の物理量の関係を表すグラフから変曲点のみを取り出して，固定小数点の配列データとして表現したものである。固定小数点マップの演算処理では，入力が与えられたとき，どの変曲点の区間に存在するかを検索し，その区間内で補間演算によって出力を算出する処理を行う。

固定小数点マップの補間演算を行う区間で入力に対する出力の変動幅が大きくなっており，補間演算の演算精度が荒くなっていることが原因であると分かった。そのため補間演算を行う区間で入力に対する出力の変動幅が必ず出力の分解能未満になるように固定小数点マップの演算処理の改善を行い，再度モデルとソフトウェアの出力比較を行った。その結果を**図6**に示す。改善後，ソフトウェアでC-ECCの出力が許容誤差1Nm以内に収まるようになった。

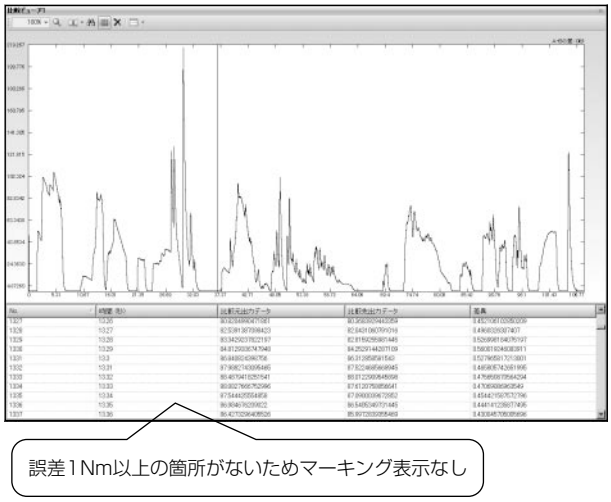


図6．許容誤差を超えるC-ECCの出力検出(改善後)

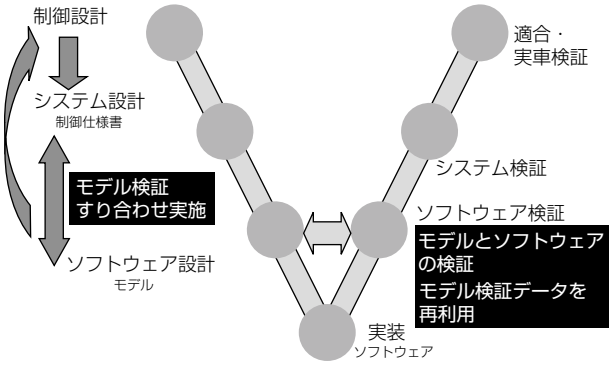


図7．モデルベース開発のプロセス改善

3. モデルベース開発のプロセス改善

モデルベース開発のプロセス改善について，**図7**に示す。**図7**は，一般的なソフトウェアの開発プロセスであるV字プロセスにモデルベース開発の各プロセスを当てはめたものである。制御設計からソフトウェア設計では，モデルのシミュレーションを繰り返し行うことによって制御仕様の検証が行われる。この制御仕様の検証で，モデルのシミュレーションに用いた入力データをソフトウェア検証で“MC-Checker”の入力データとして再利用することで，“MC-Checker”の入力データとして設計すべきテストケースの作成工数を削減でき，ソフトウェア検証の効率化が図れると考える。

4. むすび

“MC-Checker”の開発によって，モデルベース開発の検証プロセスの効率化を図ることができた。また，今後の課題として，“MC-Checker”に処理時間計測機能を追加するなど，メモリ容量以外のマイコン制約に関しても検討できるように開発を進めていきたい。