技術紹介

自律ビークル用 ECU ソフトウェアの短期開発技術について

Rapid Software Development for Autonomous Vehicle ECUs

藤井 北斗　神谷 剛志

Abstract

Yamaha Motor has been conducting research on autonomous vehicles that carry out outdoor surveying and monitoring work in place of humans (Yamaha Motor Technical Review, 2008-12 No.44, pg. 49, 2013 No.49, pg. 39). This autonomous vehicle control system comprises of an autonomous control PC that carries out host control (i.e. autonomous controls), and multiple ECUs (Electronic Control Unit) that control components such as throttle, brake and actuator based on the instruction given by the PC (see figure 1).

In autonomous vehicles research, many hours are spent on building and evaluating ECU software. For this reason, reducing the ECU software development time would be key to improving the development speed of autonomous vehicles.

This report introduces fast and efficient development technology of ECU software for the purpose of reducing the development time of autonomous vehicles.

1 はじめに

ヤマハ発動機（以下、当社）ではこれまで、屋外で人の代わりに測量や監視などを行う自律ビークルの研究を進めてきた[1][2]。当社自律ビークルの制御システムは、自律制御など上位の制御を行う自律制御 PC と、PC からの指示でスロットルやブレーキなどをアクチュエータモータによって制御する複数の ECU（Electronic Control Unit）から構成されている（図 1）。

このような自律ビークルの研究において、ECU 用ソフトウェアの実装や評価に必用な工数は決して少なくない。このことは開発期間の長期化を招き、開発コストを増大させるため、ソフトウェア開発速度の改善が切望されていた。

そこで、自律ビークル用 ECU をプラットフォーム化することで、ソフトウェアの開発期間短縮を実現した。本稿ではその開発技術について紹介する。

2 開発のねらい

図 1 に示すように、自律ビークルはアクチュエータを駆動させるための専用 ECU を複数備えている。自律ビークルの開発速度向上を目的に、これら ECU のソフトウェア開発期間を短縮したいという要望があった。

自律ビークルの ECU 開発は、①制御モデルの設計⇒②シミュレーションを用いた机上評価⇒③制御モデルのプログラミングおよびインテグレーション⇒④プログラムの実機評価といった流れで開発されるが、手作業で行っていたプログラミング作業に特に時間がかかっていた。この ECU プログラミング作業を自動化することで、ソフトウェアの開発期間を劇的に短くできると考えた。

そこで、制御技術者が設計した制御モデルから市販の自動プログラム生成ツール（ACG: Auto Code Generator）によって生成された制御アプリを実行可能な自律ビークル用「ECUプラットフォーム」と、ECUプラットフォームと制御アプリとを自動

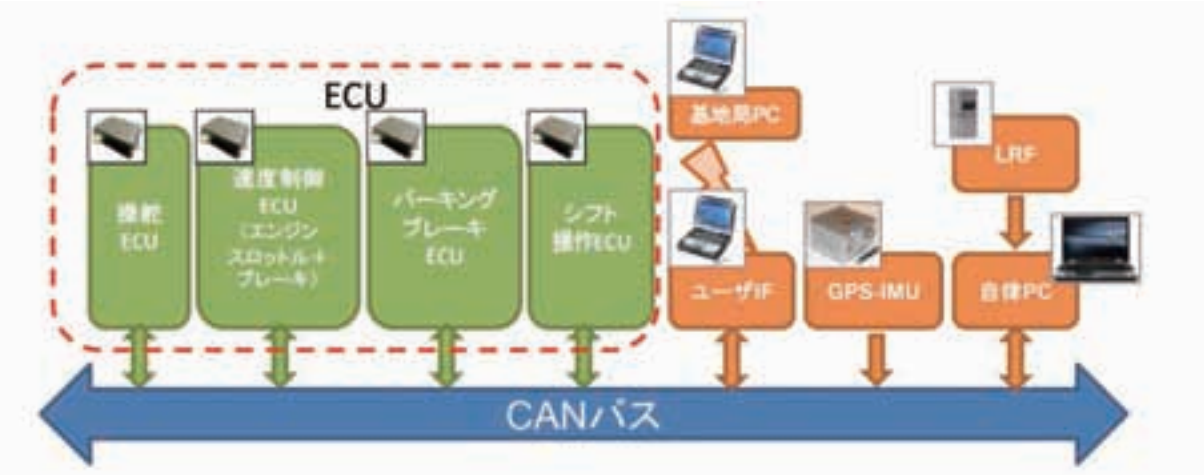


図1　自律ビークルのシステムの例

インテグレーション可能な自律ビークル用「統合開発環境」を独自に開発し活用することで、ECUソフトウェア短期開発の実現を目指した。

ECUプラットフォームと、その統合開発環境からなる短期開発技術について以下で詳細に説明する。

3 ECUプラットフォーム

ECUプラットフォームは、自律ビークルのアクチュエータ制御アプリを動作させるための実行環境であり、「ハードウェアプラットフォーム」と「ソフトウェアプラットフォーム」とによって構成される。

3-1. ハードウェアプラットフォーム

ハードウェアプラットフォーム（以下、HPF）の外観を図2に、詳細な仕様を表1に示す。

図2に示すHPFは、当社自律ビークル用に開発したオリジナルのECUである。具体的には表1に示すように、本HPFはルネサス社製SH-7047マイコンや、CAN通信やデジタル入出力などの自律ビークルでよく利用される入出力機能や、自律ビークルのアクチュエータモータを駆動するためのパワーデバイス回路などを備えた自律ビークルの車両制御に適したECUである。

図2　ハードウェアプラットフォームの外観

表1　ハードウェアプラットフォーム仕様

マイコン性能	
マイコン	SH7047（49.152MHz）
RAM	12KB
ROM	256KB
OS	TOPPERS/FI4（μTron4.0準拠）
周辺機能	
マイコン監視機能（ウォッチドッグ）	○
E-Stop（外部非常停止機能）	○
デジタル入力	8ch
デジタル出力	8ch
シリアル	1ch
A/Dコンバータ	8ch
パルス幅計測	2ch
温度センサ	2ch
CAN	1ch
モータドライブ基盤対応表	
Type1（60A仕様）	○
Type2（30A×2ch仕様）	○
Type3（30A×1ch仕様）	○
備考	アクチュエータの種類に合わせて、モータードライブ基盤を選択して使う。
開発環境	
ヤマハオリジナルMBD開発環境	○
C,C++（gcc）	○
その他	
動作電圧	12V
ケースサイズ（ドライブ基盤含む）	135×105×40 mm

3-2. ソフトウェアプラットフォーム

ソフトウェアプラットフォーム（以下SPF）の構造を図3に示す。

図3に示すように、本SPFは「システムサービス層」、「デバイスドライバ層」、制御アプリケーションとソフトウェアプラットフォームを接続するための「インターフェース層」とによって構成されており、欧州で標準化されたAUTOSAR規格を参考にしつつ研究対象である自律ビークルに最適化した当社独自の構造であることを特徴としている。

制御アプリは、永和システムマネジメント製AUTOSAR RTEジェネレータ[3]によって自動生成されたインターフェース層を介して、システムサービス層のOS時間管理機能やデバイスドライバ層のハードウェア機能を利用することができるようになっている。

図3　ECUプラットフォームの構造

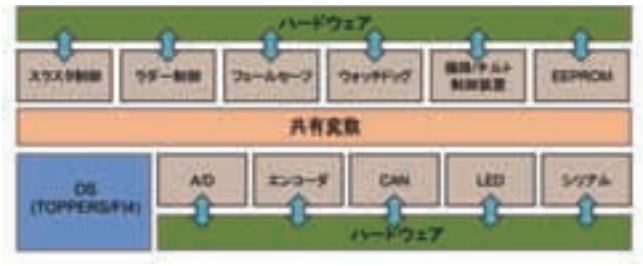


図4　従来のECUソフトウェアの構造

具体的には、過去の自律ビークル用に開発した ECU プログラムの構造（図 4）を再設計し、利用頻度の高い機能をデバイスドライバ化することで自律ビークルに適した SPF となっている。また、デバイスドライバと同様に OS も過去の ECU で動作実績のある TOPPERS FI4 カーネルを採用している。このように、本 SPF は過去の ECU ソフトウェアを再設計した構成であるため、メモリサイズ、処理負荷などが従来のソフトウェアと同等となっている。このため、開発当初から安定したパフォーマンスで使うことができる信頼性の高いプラットフォームとなっている。

4　統合開発環境

本統合開発環境は、MathWorks 社の制御モデル設計・検証ツール Matlab/Simulink を独自に拡張したものであり、図 5 に示すように「ECU ソフトウェアのモデリング」や「ECU ソフトウェアのシミュレーション」や「ACG による制御アプリの生成および ECU プラットフォームへの自動インテグレーション」などの処理が実行可能である。

本開発環境を用いることで、従来特に時間がかかった制御モデルに基づく制御アプリのプログラミングやインテグレーション作業が自動化される。その結果、ECU アプリケーションの開発時間を従来よりも劇的に短縮することが可能となった。

本開発環境の機能の詳細を以下に説明する。

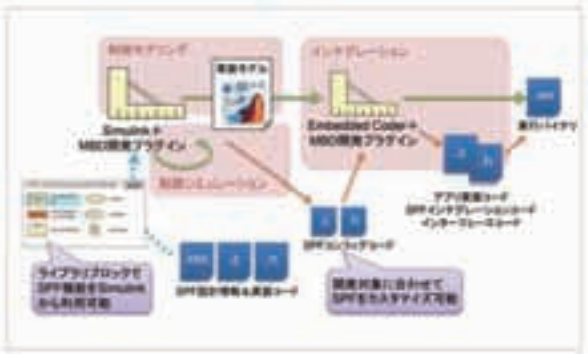


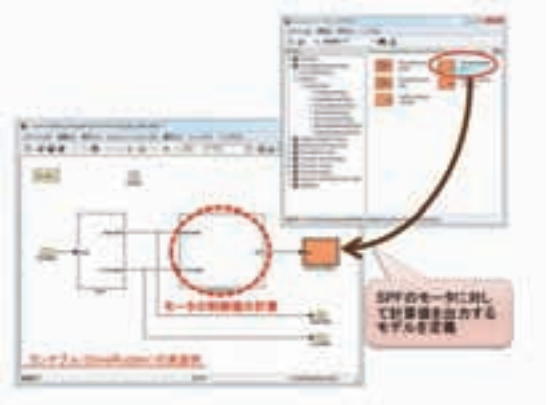
図5　統合開発環境による作業の流れ

4-1. ECU ソフトウェアのモデリング

統合開発環境は、ECU ソフトウェアの設計（モデリング）機能を備えている。このモデリング機能は、Simulink が備える制御モデルのモデリング機能を以下のように独自に拡張することで、SPF の機能を利用できる制御アプリの設計が可能となっている。

具体的には、①ソフトウェアの動作スケジューリング、②ECU プラットフォームによって提供される入出力機能（アクチュエータモータ、CAN 通信など）を制御モデルから利用するための拡張ブロックライブラリを追加した（図 6（a）参照）。この周辺機器入出力用の制御ブロックをアプリケーション処理のブロック線図上に配置するだけで、制御モデルは ECU 実装時に ECU プラットフォームの機能が利用可能となる（図 6（b）参照）。

(a) SPFブロックライブラリ



(b) SPFブロックライブラリの使い方

図6　モデリング機能

4-2. ECU ソフトウェアのシミュレーション

この統合開発環境は、シミュレーション機能（SILS: Software in the Loop Simulation) も備えており、実 ECU への実装を行う以前にシミュレーションによる机上での動作確認を行うことができる。具体的には、モデル上に配置されたモード切り替えブロックをクリックすることで、モデリングモードと SILS モードの切り替えができる。SILS モード時には、SPF 用ライブラリブロックが SILS 検証用プラントモデルとの接続端子に自動的に置き換わる（図 7）。Simulink で開発した自律ビークルのプラントモデルやテストデータをこの外部接続端子に接続することで、机上で ECU ソフトウェアの動作確認やチューニングが Simulink 上で可能となる。

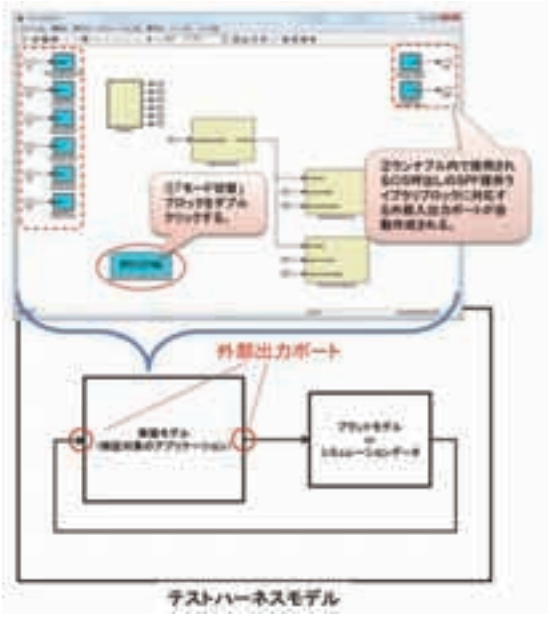


図7　SILSシミュレーション機能

4-3. ECU 実装（自動インテグレーション）

この統合開発環境は、自動インテグレーション機能も備えている。図 8（a）に示すメニューからビルドを実行すると、図 8（b）に示すように Simulink が備える ACG 機能と自動インテグレーション機能によって、制御モデルから ECU ソフトウェアの実行バイナリを生成するためのビルド処理が開始され、ただちに ECU での実機動作検証が可能となる。

具体的には、まず Simulink 上でモデリングされたアプリケーションは、Simulink が備える自動コード生成機能によって C 言語に変換される。変換された制御モデルと ECU のソフトウェアプラットフォームは、開発環境によって接続関係を解析された後、互いを接続するためのインターフェースコードが自動的に生成される。その結果、相互に接続された ECU ソフトウェアとなり、最終的に実行バイナリファイルが生成される。この実行バイナリが ECU に転送されることで、直ちにソフトウェアの実機動作検証が可能となる。

この自動インテグレーション機能により、従来時間がかかった人の手によるプログラミング作業が不要となり、ECU ソフトウェアの短期開発ができるようになった。

(a) ビルドメニュー

(b) ビルド実行画面

図8　インテグレーション機能

5 おわりに

本稿では、研究用自律ビークルの開発速度向上を目的とした ECU ソフトウェアの短期開発技術を紹介した。

具体的には、自律ビークル専用 ECU への制御モデルの自動インテグレーション技術によってソフトウェアの実装期間が大きく短縮されている。また、実 ECU 評価前にシミュレーション技術によって事前の机上評価を行うことで、実 ECU を使ったソフトウェアの評価期間も短縮されている。

今後の課題ではあるが、ハードウェアの故障など異常に対応できるソフトウェアの開発も求められる。例えば、ISO26262

などの機能安全規格に対応した OS を実装することで貢献できるのではと考えている。

■謝辞

本 ECU ソフトウェアの短期開発技術の共同開発において多大なご協力をいただいた株式会社永和システムマネジメント組込み技術センターの高橋修氏、森崇氏、中垣内勇祐氏に厚く御礼申し上げます。

■参考文献

〔1〕石山健二、神谷剛志：ロボットカーによる建設現場における無人測量、および経路追従制御のための位置・姿勢推定技術；ヤマハ発動機技報　2008-12 No.44

〔2〕平松裕二、藤井北斗、神谷剛志、望月靖之、大沼和樹：無人車開発用環境シミュレータの開発；ヤマハ発動機技報 2013 No.49

〔3〕株式会社永和システムマネジメント ホームページ「AUTOSAR RTE」〈http://www.esm.co.jp/service/service/etec_service/artec_rte.html〉（アクセス日 2014/11/12）

■著者

藤井 北斗
Hokuto Fujii
技術本部
研究開発統括部
先進技術研究部

神谷 剛志
Tsuyoshi Kamiya
技術本部
研究開発統括部
先進技術研究部