ロケット搭載ソフトウェア用検証試験ツールの進化

Evolution of tool to verify the on-board software for H-IIA/H-IIB rocket.

福田 修三*　堀田 学**
Shuzo Fukuda, Manabu Hotta

H-ⅡA/H-ⅡBロケット用搭載ソフトウェアを検証するためのツールの１つとして、フルソフトウェア・シミュレーション検証試験ツールがある。
搭載ソフトウェアの実物を検証できる新しいフルソフトウェア・シミュレーション検証試験ツールを共同開発したため、その進化した新ツールについて紹介する。

As one of the tools to verify the on-board software for H-ⅡA/H-ⅡB rocket, the full-software simulator(FSST) is essential and very important.
MSS, JAXA and NEC have jointly developed a new FSST, which can verify the real on-board software.
In this paper, we introduce this enhanced FSST and its evolution.

1. まえがき

H-IIA/H-IIBロケット用フルソフトウェア・シミュレーション検証試験ツール（FSST）は、慣性センサユニット（IMU）などの各種センサ出力、エンジン操舵などの各種モデルの動作、ロケットの物理運動などを模擬するシミュレータと、H-IIA/H-IIBロケット用搭載ソフトウェア（OBS：On-Board Software）を結合し、閉ループで動作させるツールである（図１）。

これまでのFSSTは、OBSの一部を変更し再度コンパイルしたものを供試体として検証していた。なぜなら、実ロケットのリアルタイムOS（RTOS：Real-Time Operating System）上でタスクとして動作するOBSを市販計算機の一般OS上で動作させる必要があったためである。

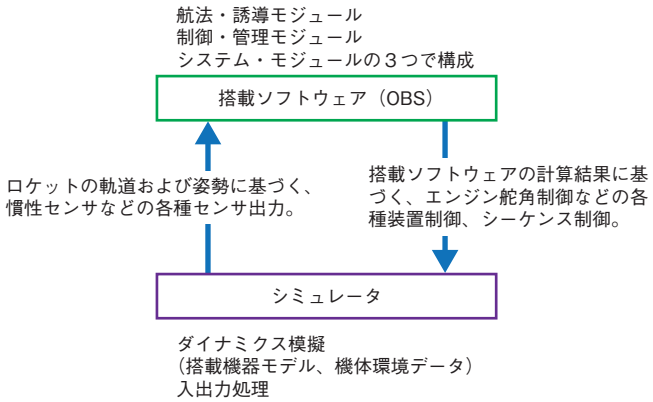

図１　閉ループによる結合・動作

当社、宇宙航空研究開発機構（JAXA）殿、日本電気株式会社殿の３社で開発した新しいFSSTによって、OBSの実物（実ロケットにロードするモジュールそのもの）を供試体とし、実ロケットと同じRTOS上で動作させ検証することが可能となった。

2. OBSとその検証について

FSSTの進化について述べる前に、FSSTの供試体である“OBS”とFSSTの目的である“OBSの検証”について述べる。

OBSは、制御・管理モジュール、システム・モジュール、そして当社が担当し開発した航法・誘導モジュールの３つのモジュールから構成される。航法・誘導モジュールはIMUからの情報を基にまず航法計算としてロケットの現在位置・現在姿勢を把握し、次に誘導計算として目標地点に到達するための機体回転レートやエンジン停止時刻などを計算する。制御・管理モジュールは航法・誘導モジュールの計算結果や各種機体情報等を基にロケット機体の制御を行う。システム・モジュールは制御・管理モジュールによるロケット制御を実現するために誘導制御計算機（GCC）との入出力の仲介（I/Oインタフェース）を行う。これら３つのモジュールが連係して動作することにより、地上からの遠隔操作に依らない自律飛行が可能となる。

OBSの検証は、開発時と運用時（打上号機毎の定数を適用してのミッション適合性を含めた検証が実施され

＊つくば事業部 第一技術部　＊＊つくば事業部 中部分室

る）で異なる点は多々あるものの大まかには３つに分類できる。まず各モジュール毎に単体検証、機能検証、統合検証が実施される。次に、各モジュールを結合したOBSと誘導制御系機器（実ハードウェア）を組み合わせた誘導制御系システム試験が実施される。そして最後に、フルソフトウェア・シミュレーション検証試験（以下、「FSST検証試験」と称す）として、各モジュールを結合したOBSを適用しての実ハードウェアを伴わないソフトウェアのみでの検証が実施される。

3. FSSTの用途

OBSの各モジュールを結合した検証は、２章で示したとおり、誘導制御系システム試験およびFSST検証試験で実施される。

誘導制御系システム試験はOBSと誘導制御系機器（実ハードウェア）を組み合わせての試験であるため、実ハードウェアの異常・故障ケースの実行には制限がある。また試験規模が大きいため手軽に実施できない。よって、異常・故障ケースをソフトウェアのみで手軽に実施できるFSST検証試験が実施されている。

FSSTは実ハードウェアを伴わないソフトウェアのみでの検証ツールであるため、実ハードをソフトウェアでモデル化できれば、多様な事象に基づくOBSの動作を手軽に検証することが可能となる。

4. FSSTの進化

H-IIA/H-IIBロケット用の新GCCとしてマイクロプロセッサHR5000[3]が適用されたことにより、2010年度から2011年度にわたってFSSTを作り変えることとなった。これまでのFSSTではOBSの実物ではなく市販計算機用に修正／コンパイルしたものを適用していたため、OBSの実物を適用できるようにHR5000エミュレータを導入しFSSTを進化させることとなった。

HR5000エミュレータとはGCCをソフトウェアで模擬したものである。HR5000エミュレータ上でOBSの実物を動作させることにより、実機上で動作させた場合と同じデータを取得可能となる。OBSの実動作を検証できれば信頼性は大きく向上する。

なお、HR5000エミュレータはあくまでGCCの模擬であるため、実機におけるGCCと実ハード間の入出力（I/O）データおよび実ハードのモデルは別途準備しなければならない。このI/Oデータおよび実ハードのモデルはシミュレータ（図１）が担っている。

5. 新旧FSSTの相違

１章および４章で述べたとおり、新旧FSSTの大きな相違はOBSの実物を使用しているか否かである。その相違により新旧FSSTの構成は大きく異なるものとなった。

旧FSSTは、シミュレータの内側にOBSとのインタフェース（I/F）機能を作成し、そのI/F機能の内側に市販計算機で動作するように修正／コンパイルしたOBS を組み込んでいる（図２）。シミュレータ、I/F機能、OBSは共有メモリ下に置かれている。コンパイルしなおしたOBSを利用するためエミュレータ部分は存在しない。

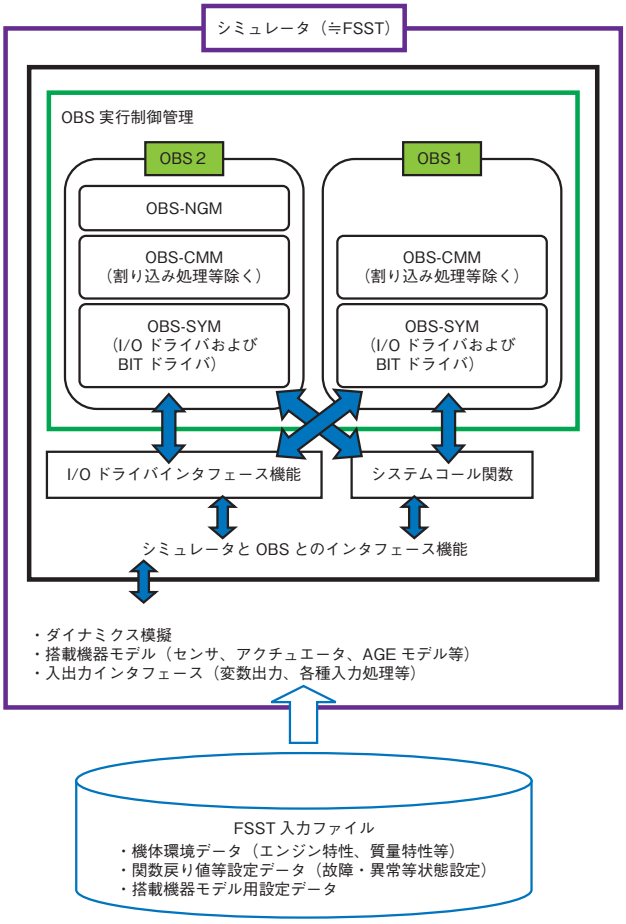

図２　旧FSSTの構成

新FSSTは、機体運動模擬部（FSST-SIM）と称するシミュレータ部分と誘導計算機模擬部（FSST-EML）と称するエミュレータ部分が完全に分離しており、両者間のデータのやり取りはTCP/IP通信で行っている（図３）。新FSSTでも共有メモリを使用したI/Fは可能であったが、検討の結果、TCP/IP通信を採用した。その主な理由は、TCP/IP通信の方がインタフェース分解点が明確でありFSST-SIMとFSST-EMLを別々に開発しやすく且つ問題発生時に問題点を明確にしやすいこと、FSST-

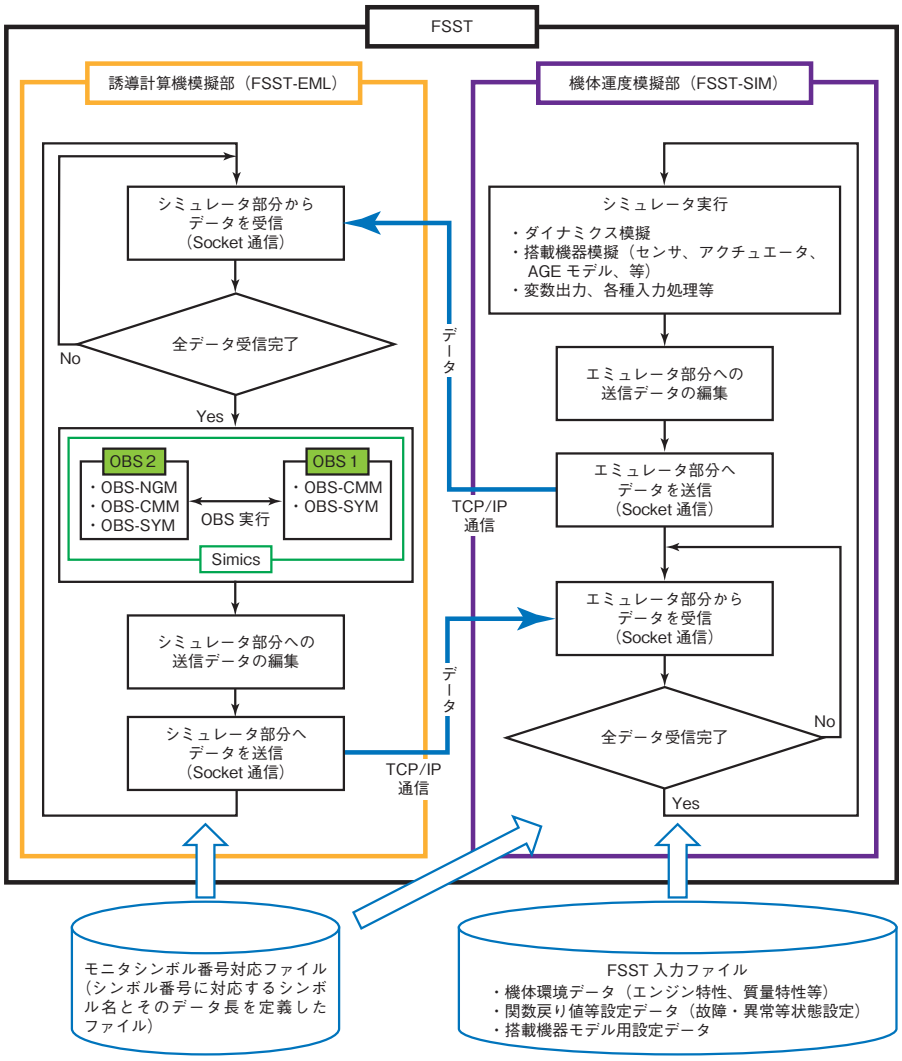

図 3　新FSSTの構成

SIMは32bit環境であるのに対しFSST-EMLは64bit環境であるため、共有メモリとした場合はFSST-SIMの開発／検証規模が大きくなると予想されたことである。

なお、TCP/IP通信の場合は開発時には便利であるが運用時には解析データを容易に確認できないというデメリットもある。例えばOBS内のある変数の値を利用したいという場合、共有メモリの場合はグローバル変数として定義することで容易に利用可能であるが、TCP/IP通信の場合はFSST-SIMとFSST-EMLの両者に該当変数が何かを共通認識させると共に、該当変数の値を通信データに追加しなければならない。新FSSTではこの欠点を通信データを工夫することで緩和している。詳細は７章で述べる。

6. 新FSSTの構成（誘導計算機模擬部と機体運動模擬部）

FSST-EMLは、Simics [4]（ハードウェアのシミュレーションを行う仮想プラットフォーム）のHR5000エミュレータ機能にGCCをエミュレートする機能およびFSST-SIMとのTCP/IP通信機能を付加したプログラムである。FSST-SIMは、当社がJAXA殿との契約のもと開発／納入した飛行経路計算プログラム（ALMA：Analyzer for Launch vehicle Motion and Attitude）を改修し、FSST検証試験に必要なモデルやFSST-EMLとのTCP/IP通信機能等を追加したプログラムである。このFSST-EMLとFSST-SIMを同期させ動作させることにより閉ループ解析（図１）が可能となる。

FSST-EMLの主な役割は、FSST-SIMから受け取ったダイナミクス・データを基にOBSを動作させることである。FSST-SIMの主な役割は、FSST-EMLから受け取ったOBSの指令を機体ダイナミクスに反映することである。

なお、FSST-EMLは日本電気株式会社殿がJAXA殿との契約のもと開発し、FSST-SIMは当社がJAXA殿との契約のもと開発した。

7. FSST-EMLとFSST-SIMとのインタフェース

FSST-EMLとFSST-SIM間のI/Fデータは TCP/IP通信でやり取りしているため、I/Fデータの仕様は、開発／検証の容易性を優先し設計した。

I/Fデータのレコード長は、モニタ用の変数データのみを可変長とし、その他のデータは全て固定長とした。ただし、FSST-SIMの仕様により可変長データの大きさはシミュレーション開始時に確定するため、シミュレーション中は可変長データについてもデータ量は一定であり固定長と同様に扱うことが可能となっている。バイトオーダーは適用する計算機が確定しているためlittle endianに固定とした。I/Fデータのファイル・フォーマットは全てに共通で表１のとおりとした。表１中のデータのタイプとは、設定に関するデータ、変数の値等々の送受信データの種類を識別するためのものである。

新FSSTではOBSの実物を使用しているためOBSの変数値を確認したい場合、その変数のアドレスを把握しダンプしなければならない。しかもOBSが改修された場合はアドレスが変化する可能性があるため、アドレスを固定にすることはできない。さらに、TCP/IP通信でFSST-SIMに変数値を送信するため、送信データが何の変数値なのかをFSST-SIMとFSST-EMLで共通に認識する必要がある。

新FSSTでは変数に対応する番号（モニタシンボル番号）を定義しその番号をFSST-SIMからFSST-EMLに送信することでモニタしたい変数が何かを共通認識できるようにしている。具体的変数名ではなく番号としたのは、データサイズを固定にできることと通信量が少なく済むためである。FSST-EMLはモニタシンボル番号に対応する変数のダンプを取得し物理量に変換しモニタシンボル番号と同じ並びでFSST-SIMに送信する。ダンプする際のアドレスはOBSの各モジュールをリンクした際のMAPファイルより取得する。モニタシンボル番号と変数の対応はモニタシンボル対応ファイルで行っている。モニタシンボル対応ファイルにはシンボル番号に対応する変数名とそのデータ長を定義している。このモニタシンボル対応ファイルはFSST-SIMとFSST-EMLの両方に外部ファイルとして与える。以上の処置を行うことで、FSST-SIMからFSST-EMLに送信するモニタシンボル番号さえ変更すれば、手軽に変数の値を確認することが可能となる（図４）。

その他の代表的なI/Fとしては、OBS定数を変更して検証する場合への対応としてOBSへのパッチ機能などがある。

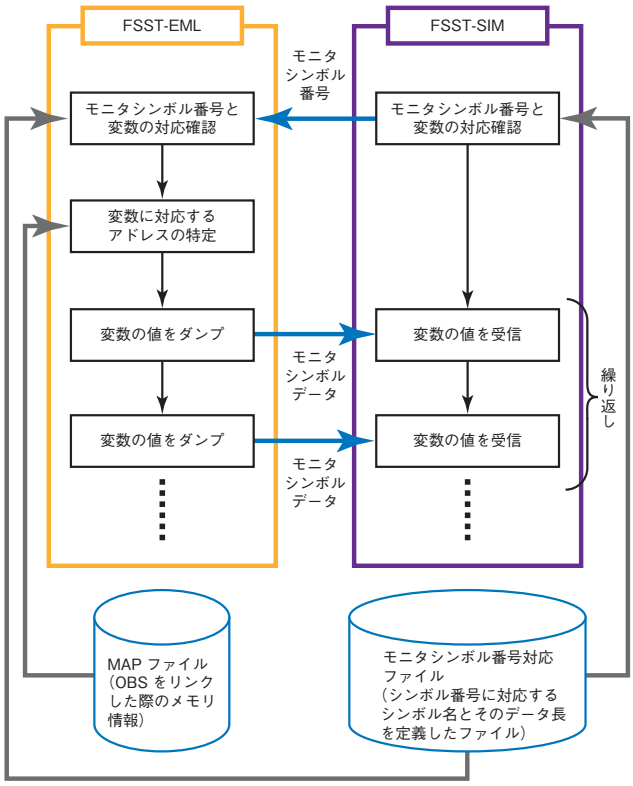

図４　変数のモニタ方法

8. 工夫点

特殊な工夫の代表例として、OBSが参照した定数値をモニタする方法を紹介する。

例えば、OBSが構造体配列で定義された定数を構造体へのポインタから間接参照している場合、グローバル変数を使用していないことからモニタシンボル番号（７章）によるアドレス特定ができずOBSが取り出した定数値をモニタすることはできない。なぜなら、ポインタをモニタしてもFSST-EMLとFSST-SIMではメモリ空間が異なるため、ポインタの先にある情報が異なり意味を持たないためである。

よって、モニタしたい定数を含む構造体配列の全体を一度モニタした後に、OBS内でのポインタ変化量を“モニタした構造体配列全体”に適用し、該当定数を特定し取り出すという処置を行っている。しかしこの場合、構造体配列のサイズが大きいとすぐに通信量の制限（シミュレーション時間に影響するため通信量を制限している）

表１　I/Fデータのファイル・フォーマット

ヘッダー部（64ビット）(バイトオーダーはlittle endianに固定)			データ部（バイトオーダーはlittle endianに固定）
予備（28bit）	データのタイプ（４bit）	データのバイト数（32bit）	データ（サイズはヘッダー部の指定にに従う）

にかかってしまい他の変数を一切モニタすることができなくなる。よって、定数値については別途ソースファイルをコンパイル・リンクし、そのモジュールから値を取り出すことも可能とした（図５）。定数値をI/Fデータからモニタするか、別途コンパイル・リンクしたモジュールからモニタするかはFSST-SIMの設定で簡単に切り替えることができる。

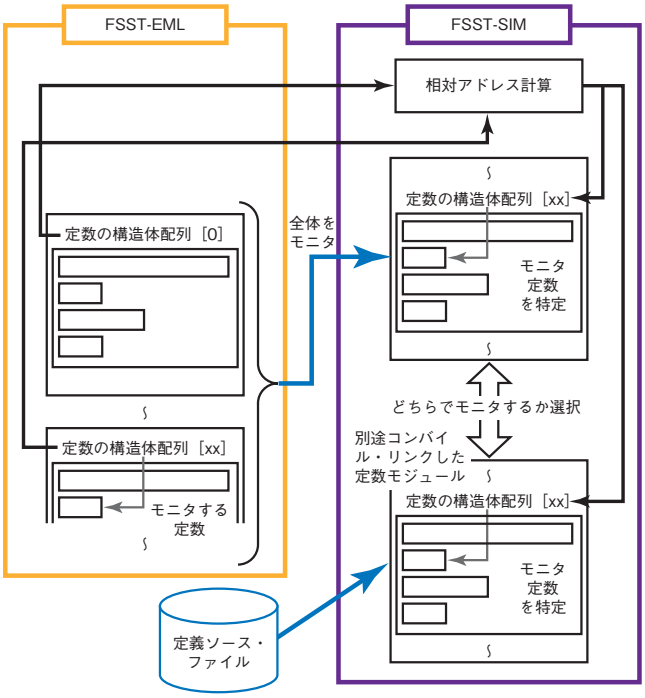

図５　テーブル定数のモニタ方法

9. 将来の課題、展望

市販の高スペックCPU（Xeon® X5690 3.46GHz）を使用しても新FSSTのシミュレーションに要する時間は実時間の約1.8倍である。8,000秒のシミュレーションを実施した場合、シミュレーション終了まで約14,400秒（約４時間）かかる計算になる。時間を要する原因はエミュレータを使用していることによる。新FSSTではなくALMA（飛行経路計算プログラム）にて8,000秒のシミュレーションを行うと数十秒程度で終了するため、エミュレータの負荷がいかに大きいかがわかる。シミュレーション時間の改善が今後の課題である。

将来の展望としては、誘導制御系システム試験の一部をFSSTで代行できる可能性があることである。代行できればロケット開発時およびロケット運用時のスケジュールの短縮、そして信頼性の向上に貢献できるものと思われる。

10. むすび

今回の新FSSTの開発をとおして、色々な問題点や課題を認識することができた。これら問題点や課題を今後のFSSTの改善に活かし、OBSの検証およびロケットの開発・運用に貢献したいと考えている。

ロケットは大変高価なものである。その高価なロケットの打上げをソフトウェアの不具合に起因して失敗すれば、それは痛恨の極みである。なぜなら検証により問題点を解決できていれば防ぐことができたに違いないからである。ソフトウェアに対しては今まで幾重もの検証を実施し信頼性を高めてきた。この方針は今後も変わらないであろう。故にOBSの実物を検証できるFSSTは今後更に重要性を増す可能性がある。今後もFSSTの技術の維持・発展に努めていきたい。

参考文献

(1) 第55回宇宙科学技術連合講演会論文：2D06 H-IIA/H-IIB ロケット用搭載ソフトウェア検証ツールの開発（横田清美、泉達司、石濱直樹、福田修三、堀田学、渡辺仁志、下村知子）

(2) NEC技報：http://www.nec.co.jp/techrep/ja/journal/g11/n01/110125.html
ロケット用誘導制御計算機の変遷と展望（林　伸善）

(3) HR5000：http://www.hirec.co.jp/eng/business/pdf/320MIPS64bitMPU.pdf

(4) Simics：http://www.windriver.com/japan/products/simics/index.html