Model Predictive Control real time performance in a rendezvous&capture scenario for Mars Sample&Return

M. Saponara, C. Philippe**, A. Tramutola**

** Thales Alenia Space Italia, Strada Antica di Collegno 253 – 10146 Torino (Italy)*
email: massimiliano.saponara@thalesaleniaspace.com
antonio.tramutola@thalesaleniaspace.com
*** ESA/ESTEC, Keplerlaan 1- 2200 AG Noordwijk (The Netherlands)*
email: christian.philippe@esa.int

Abstract
Real-time performance of a rendezvous and capture Guidance, Navigation and Control (GNC) solution based on Model Predictive Control (MPC) have been evaluated in the frame of the ESA study "On-line Reconfiguration Control System and Avionics Architecture" (ORCSAT). An Avionic Architecture Demonstrator embedding the MPC control system has been developed in this study, with the peculiarity to host a processor – co-processor configuration able to cope with the high computational capability requested by the optimization algorithms used by the MPC. The software architecture was adapted to manage the two processors and their data exchange. Comparison between simulation results with a full software environment (FES) and the Demonstrator showed minor differences and put in evidence the reconfiguration capability of the MPC in case of unexpected events.

1. Introduction

The Mars Sample&Return (MSR) mission of the European Space Exploration Programme "Aurora", has the main objective in bringing back to the Earth a sample of Martian soil. To achieve this, a rendezvous and capture system able to autonomously detect, approach and capture the sample previously put in a predefined orbit by the Mars Ascent Vehicle (MAV) is required. Fundamental parts of this system are the Guidance, Navigation and Control algorithms (GNC), that have to cope with poorly cooperative target and operational constraints. A previous ESA study "High integrity Autonomous RendezVous and Docking control system" (HARVD) led by GMV developed an automated rendezvous and capture control system based on classical control techniques [1] and [2].

The ESA study "On-line Reconfiguration Control System and Avionics Architecture" (ORCSAT) addressed the application of optimization-based control strategies such as Model Predictive Control (MPC) in the HARVD GNC. Indeed, the capability to include performance goals, optimal path planning and dynamic safety margin in an optimization problem in addition to the feedback stabilization has been considered extremely attractive for this kind of mission.

The ORCSAT study has first addressed the MPC design in a full software simulation environment (Functional Engineering Simulator (FES)). The designed solution has been widely tested by means of a Monte Carlo simulation campaign, showing great robustness against different dynamic conditions and performance improvements with respect to classical GNC solution both in terms of propellant consumption but also in terms of optimal trajectory planning [3]. The design phase have also identified that a distributed architecture for the Central Data Management Unit (CDMU), which considers a processor plus a coprocessor, is necessary to cope with high computational capability required by optimization algorithms embedded in MPC.

The final objective of the ORCSAT study was to setup an Avionics Architecture Demonstrator embedding MPC control system for rendezvous and capture scenario, in order to evaluate the performances of the developed algorithms into a space representative avionic platform. The paper will present the ORCSAT Demonstrator design and setup, with particular focus on the flight segment architecture, including GNC partitioning, implementation and testing. Then, real-time simulation results will be showed with the objective to compare them with the FES one, justifying eventual differences.

2. The ORCSAT demonstrator

2.1 Overview

The ESA study ORCSAT addressed the application of MPC techniques to the rendezvous and capture scenario foreseen for the Mars Sample&Return mission. MPC based control system has been designed, tested and validated against the reference HARVD control system using a Functional Engineering Simulator (FES) in MATLAB®/Simulink®/ environment. Then, a representative flight-like avionic architecture system (Demonstrator), allowing the implementation of embedded MPC control system, has been designed and tested in a real-time environment. The scope of Avionics Demonstrator can be summarized as follows:

- Implement the avionic architecture of the ORCSAT system using, as much as possible, flight representative components;
- Verify and validate the software architecture of the ORCSAT On-Board Software (OBSW) in a real time environment;
- Evaluate functional performances using Hardware-In-the-Loop (HIL) and dedicated facilities.

The Demonstrator consists of the following segments and environments:

- Flight Segment;
- Ground Segment;
- Real Time Simulation Environment;
- Software Development Environment.

The block diagram representing the conceptual design of the Avionics Demonstrator is shown in Figure **1**.

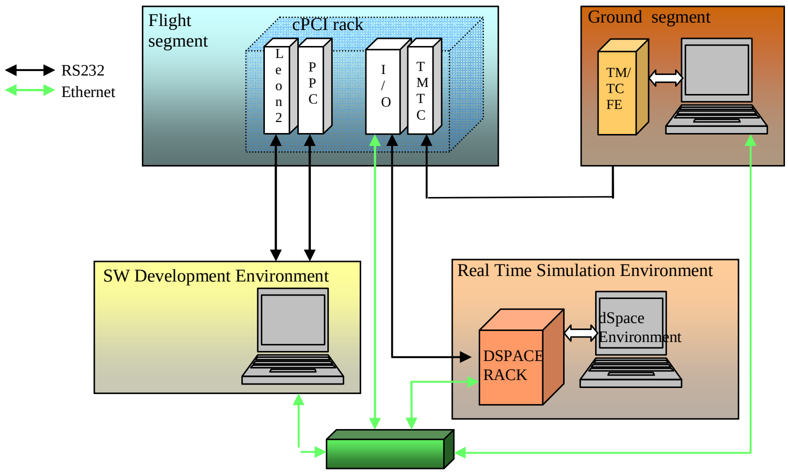

Figure 1 : Avionic Demonstrator block diagram

2.2 Flight segment

2.2.1 Hardware architecture

Flight segment HW architecture is based on Reference Avionics System Testbed Activity (RASTA) system from Aeroflex Gaisler which has been customized for the ORCSAT study. The customised RASTA system, Figure **2**, consists of a cPCI rack U3 format in which the following boards have been integrated:

- GR-CPCI-AT697 from Aeroflex Gaisler: main processor board based on Atmel AT697E (LEON2-FT);
- CPCI-750 from ESD: co-processor board based on IBM PowerPC750 FX;
- GR-RASTA I/O from Aeroflex Gaisler: Input/Output board based on Xilinx Virtex-4 FPGA hosting all needed communication interfaces;

- GR-RASTA TMTC from Aeroflex Gaisler: telemetry and telecommand interface based on Xilinx Virtex-4 FPGA implementing CCSDS TM encoder and CCSDS TC decoder.

The master of the cPCI rack is the LEON2 processor (AT697) that is able to communicate with all the other boards. As the PCI arbiter of the AT697E can manage only 3 additional PCI agents, a PCI bridge has been included inside the rack. All boards are inserted in a 21 slot air cooled cPCI crate which allows the integration in the flight segment of additional boards (i.e. FPGA boards embedding specific algorithms, mass memory cards) and interfaces (RS-232 mezzanine boards) for test purposes.

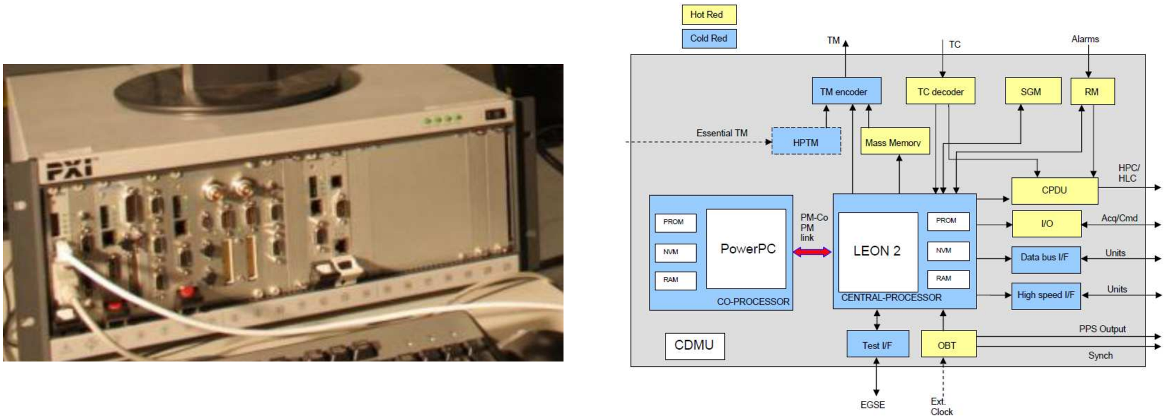

Figure 2 : Customised RASTA rack (left) and CDMU configuration (right)

As shown on Figure **2**, the main peculiarity of the ORCAST avionic architecture is the presence of a CDMU embedding a processor (LEON2) and a co-processor (PowerPC). Indeed, the MPC concept is based on the optimization of a cost function under some constraints, which usually is carried out using quite complex iterative algorithms, requiring high computational capability. Therefore, the designed architecture considers a processor devoted to data handling, navigation, safety monitoring, mission management, etc. and a co-processor fully dedicated to the optimization algorithms [4].

2.2.2 Software architecture

As described in [1], one MPC controller for each rendezvous phase has been selected as the most efficient solution for the proposed scenario:

- Orbit Synchronization Translational Guidance (OSTG), which has the objective to bring the chaser spacecraft in the same orbit of the target at an in-track separation comprised between 5 and 30 km;
- Impulsive Nominal Translational Guidance (INTG), which has the objective to progressively reduce the relative distance to the target by means of intermediate holding points up to an in-track separation of 100m;
- Forced Terminal Translational Guidance (FTTG), which performs a straight-line trajectory from 100m to 3m from the target, with the following capture accomplished in free-flight;
- Collision Avoidance Manoeuvre (CAM), which brings the chaser to a safe distance in case a collision risk is detected.

The MPC design activity has identified the Programming (LP) solver based on Dual Simplex and Quadratic Programming (QP) solver based on Dantzig algorithm required the highest computational capability. and then as the natural candidates for the implementation in the PowerPC. These algorithms are typically “nested” in the MPC design, and their extraction from the MPC controller blocks would have presented several difficulties, taking into account that each controller requires a different optimization problem in terms of size, complexity, etc. Then, it was decided to harmonize the MPC controllers interfaces (see Figure **3**) in order to put them in a dedicated block (MPC_container) that is executed on the PowerPC, defining also two interface blocks (MPC_IF_IN and MPC_IF_OUT) that run on the LEON2 (see Figure **4**). In this way a clear separation of what has to run on the different processors is obtained without ambiguities.

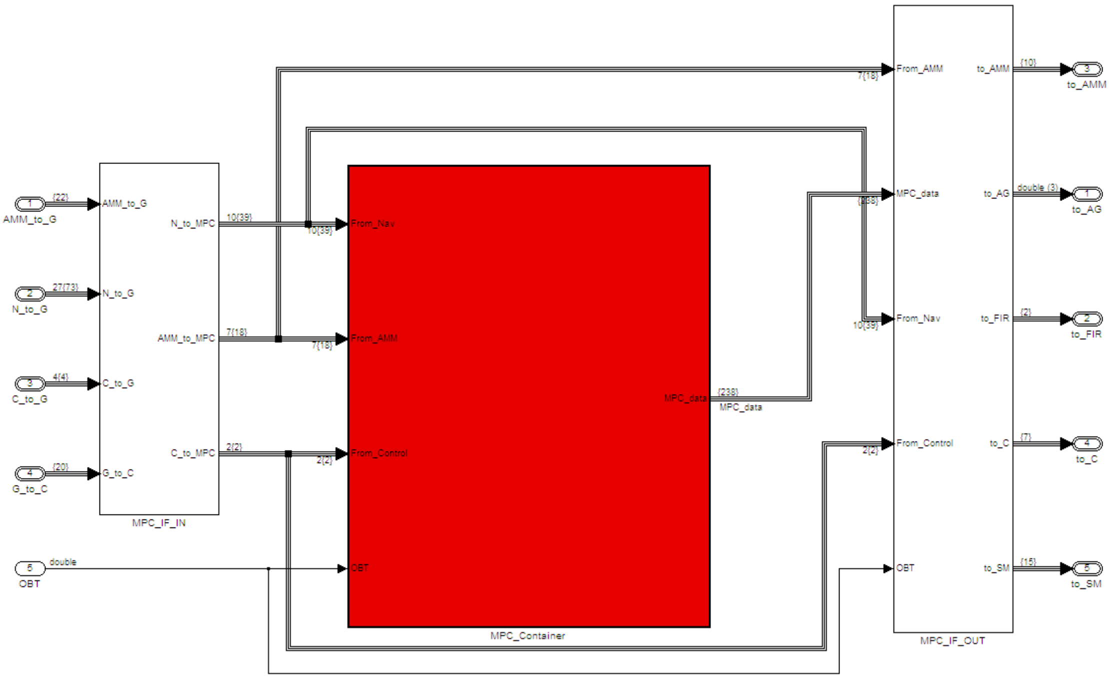

Figure 3 : MPC_Container interfaces

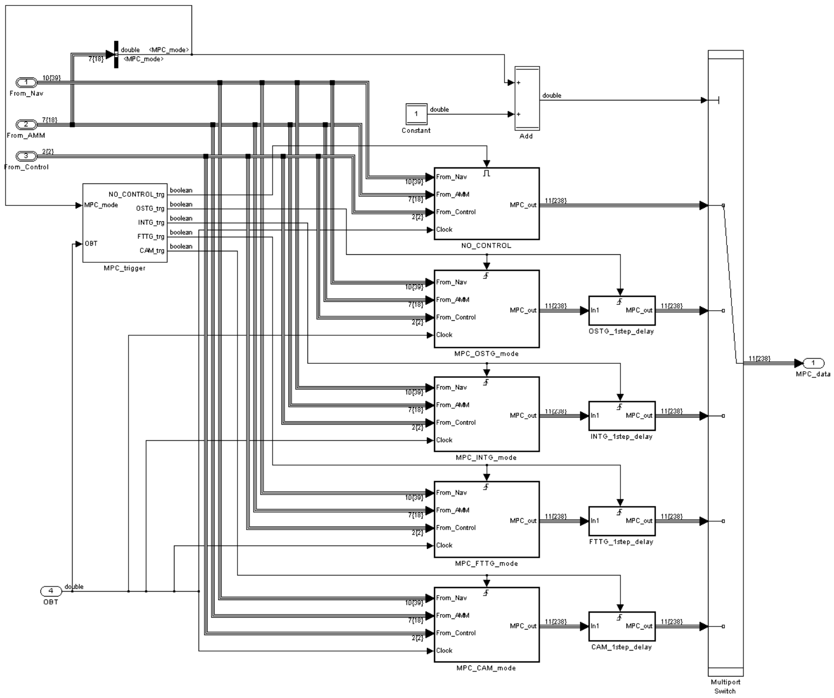

Figure 4 : MPC_Container internal layout

The next step of the design was to define the data exchange mechanism between processor and co-processor. Exploiting the discrete nature of the control, it was possible to limit the interactions between LEON2 and PowerPC at prescribed times, and in particular at each control sample step, in the following way:

- At $t = t_0$ the processor provides to the co-processor all the information (navigation, commands, etc) necessary to the optimization computation and give the command to start the computation;
- At $t = t_0$ the co-processor receives this command and starts the computations “in parallel”, then without further interactions with the processor;

- Once the calculations are finished, the co-processor results are stored in its interface toward the processor (output port);
- At $t = t_0 + t_s$, with t_s as the sample time of the MPC controller, the processor takes the output of the co-processor from this port and provide the new information for the following optimization.

This mechanism introduces a one-step delay in the actuation, since the computed maneuver with the information sampled at t_0 will be applied at $t_0 + t_s$, but since this delay is deterministic and known it can be easily accommodated in the control design. This is the simplest way for processors communication from the conceptual point of view, and the main issue is to guarantee to obtain a solution of the optimization within the allocated time. The MPC design have first defined the optimal sample time of the controller on the basis of the achievable performance, then the algorithms profiling have been carried out. Such information has been then used for the selection of the co-processor from the available candidates [3] and [4].

A model-based development approach was used throughout the entire ORCSAT study, combined with automatic C-code generation. The GNC algorithms have been designed with MATLAB®/Simulink®/Stateflow® and the overall block diagram has been organized in such a way that the auto-coded software is partitioned in two applications: one for LEON2 (all SW including GNC and MPC management) and the other one for PowerPC (MPC container). The two software modules have been integrated inside the complete On-Board Software (OBSW).

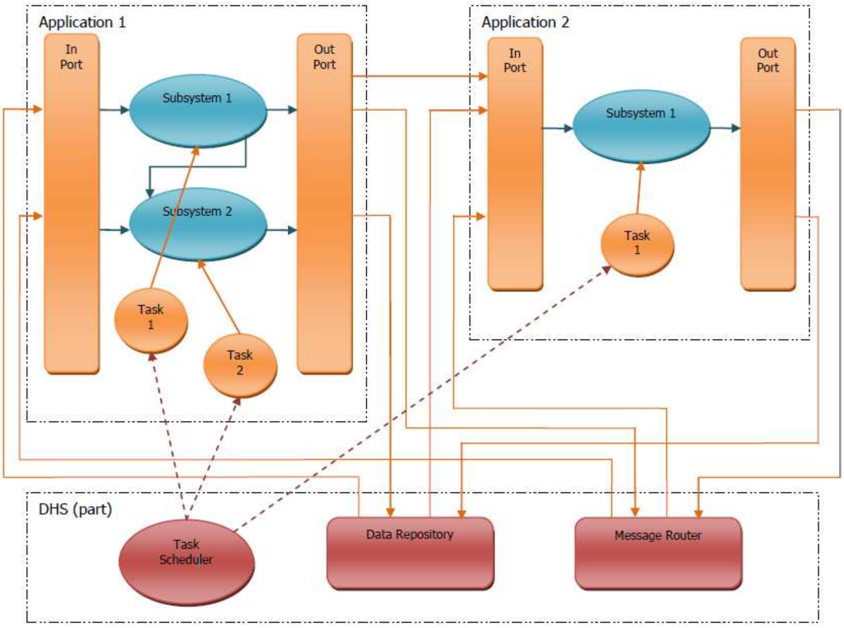

Figure 5 : Application software architecture

The OBSW of the flight segment is composed by the following elements:

- The Data Handling System (DHS), which is the generic part of the OBSW implements basic services, PUS services, boot services and system management;
- The Real Time Operating System Interface (RTOS_IF), which is a generic interface to the operating system (VxWorks 6.7 has been used in ORCSAT);
- The Kernel, which is the low level layer accessing the hardware;
- The Platform specific applications such as equipment access, power management, RF management, GNC application software, etc.;
- The Mission specific applications such as payload management and mission specific application software;
- Various utilities used by the other elements.

The OBSW is organized by means of an application framework responsible for the data handling and interface with the GNC application and the rest of the world (software, hardware and ground). The interface of these applications consists in input and output ports (Figure **5**). The role of the application framework is to ensure that the input and output ports are processed in due time. The input and output data ports are data structures that clearly materialize the inputs and outputs of the software components. A software component consumes data from its Input Port and produces data through its Output Port.

2.2 Ground segment

The ground segment consists of the following elements (Figure 6):

- CCSDS/ECSS Telemetry (TM) and Telecommand (TC) Electrical Ground Support Equipment (EGSE);
- RS422 link to interface the RASTA system TM/TC board ;
- A dedicated computer hosting a mission control system SW

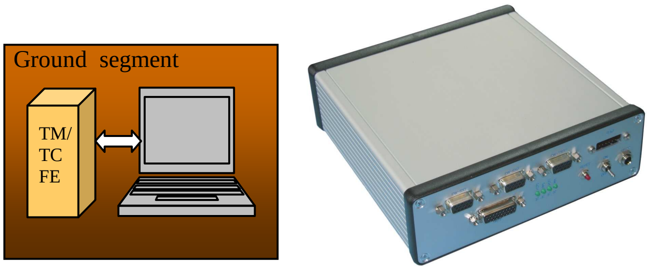

Figure 6 : Ground segment block diagram (left) and TM-TC EGSE box (right)

The CCSDS / ECSS Telemetry and Telecommand EGSE is an Aeroflex Gaisler product which communicates directly with the on-board telemetry encoder and telecommand decoder devices present in the TM/TC board of the Flight Segment bypassing the transponder. The TM/TC EGSE is fully compliant with the latest CCSDS recommendations and ECSS standards and implements the lower CCSDS / ECSS protocol levels.
The TM/TC EGSE hardware interfaces the on-board flight segment (TM/TC board) via a standardized RS422 interfaces supporting up to 10 MBPS transfers on the downlink and up to 1 MBPS on the uplink. The EGSE hardware communicates with the computer hosting the ground segment software through a 10/100 Mbit/s Ethernet interfaces via TCP/IP.
The Ground segment is completed by the RAMSES software (Rocket and Multi-Satellite EMCS Software) from Swedish Space Corporation, running in a Window XP environment, which license has been procured from Aeroflex Gaisler together with the TM/TC EGSE box. RAMSES System consists of several applications (Anubi, Cheops, Hathor, Nefertiti, Osiris, Sphinx) able to store, to process and display telemetry and to generate telecommands and procedures.

2.4 Real-time simulation environment

The real time simulation environment is inherited from the HARVD study. The Real Time Simulation Environment consists of an HW platform and a SW platform, as depicted in Figure **7**.

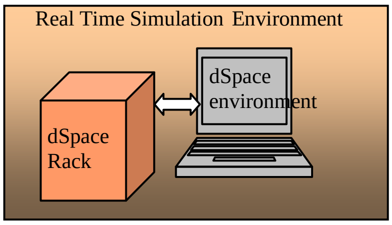

Figure 7 : Real-time simulation environment block scheme

The HW platform consists of a Development PC and the dSPACE rack. The PC drives the dSPACE board via an optical link using a dedicated DS817 board included in the PC. Serial and Ethernet connections link the PC to the other segments.
The dSPACE rack is an Expansion Box in which the following boards are assembled:

- DS1006 board: it is a quad-core AMD Opteron processor board programmable from Simulink;
- DS814 board: it is used for the communication with the development PC;
- DS4504 board: it is used for the Ethernet Network Link.

The SW platform consists in software suite, in Windows OS environment, installed on the Development PC:

- MATLAB/Simulink/Stateflow;
- Real Time Workshop (RTW);
- dSPACE TargetLink;
- dSPACE RTI/Control Desk.

It is used to perform:

- conversion from the Simulink On Board model to C code using TargetLink;
- conversion from the Simulink Real World model to the executable code using RTW;
- control and monitoring of the execution of the simulation on the dSPACE board via the Control Desk.

The executable SW loaded on the dSPACE DS1006 board is the Real Word SW simulating dynamics, kinematics, sensors and actuators. The Ethernet link driven by the DS4504 board transmit to the OBSW loaded on the flight segment processors all the interface data (sensors measurements and actuators commands). The protocol used for the data exchange is the UDP/IP.
Particular attention has been put on the synchronization mechanism between dSPACE and RASTA, in order to implement a real-time simulation. It is performed as follows:

- The RASTA sends the data via Ethernet link to dSPACE and wait for data coming from it.
- Each 100ms, the dSPACE send the data to RASTA and extract received data from a cue, which contains the last received packet

At the beginning of each simulation, dSPACE send a first data packet (dummy), synchronizing itself with the RASTA. The Simulink scheme which implements this mechanism is showed in Figure **8**.

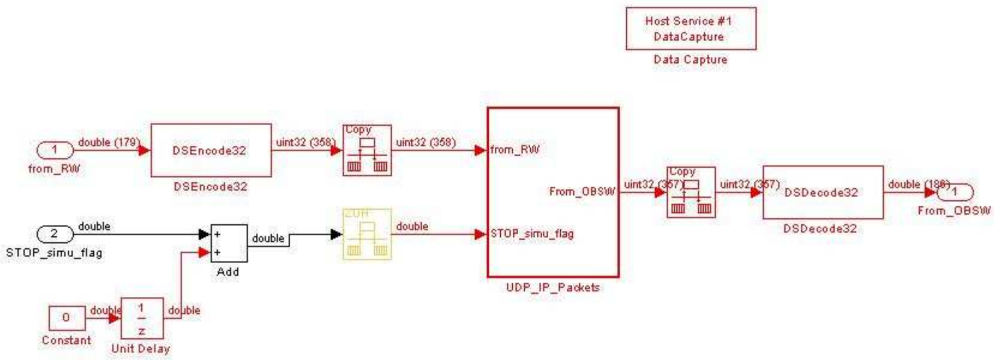

Figure 8 : Simulink scheme of dSPACE – RASTA communication interface

2.5 Software development environment

This environment consists in a dedicated computer interfacing the two processor boards of the flight segment. The functionalities implemented by this component of the ORCSAT system Demonstrator are:

- functional simulation capability (MATLAB/Simulink environment);
- automatic generation of the C code (Real Time Workshop, TargetLink);
- provision of the configuration control of the flight SW (svn);
- Testing of the flight SW using automatic tool;
- Running and debugging of the flight SW (WindRiver Tools);
- Uploading of the SW into the processor and coprocessor of the flight segment (WindRiver Workbench based on Eclipse IDE).

This environment interfaces the computer boards (AT697E and CPC750) via RS232 and Ethernet links and is linked via Ethernet to the other components of the ORCSAT demonstrator (Real-time simulation environment and Ground Segment).
The selected personal computer is an HP Z400 Workstation with Intel Xeon W3580 3.33 8MB/1333 Quad Core CPU and 4GB RAM. The computer is equipped with a 500GB SATA 7200rpm HDD. Operating system installed is Windows XP 32Bit. As this computer is not equipped with serial ports an USB to Serial converter has been procured to interface the two microprocessor boards.

3. Real-time simulation results

3.1 Test 1: circular orbit

Figure **9** shows the comparison between the trajectories and the final capture accuracy obtained with the different simulation environments (FES and Demonstrator, the latter named in the pictures “OBSW”), in a scenario considering a circular rendezvous orbit with the chaser placed at about 300km from the target.

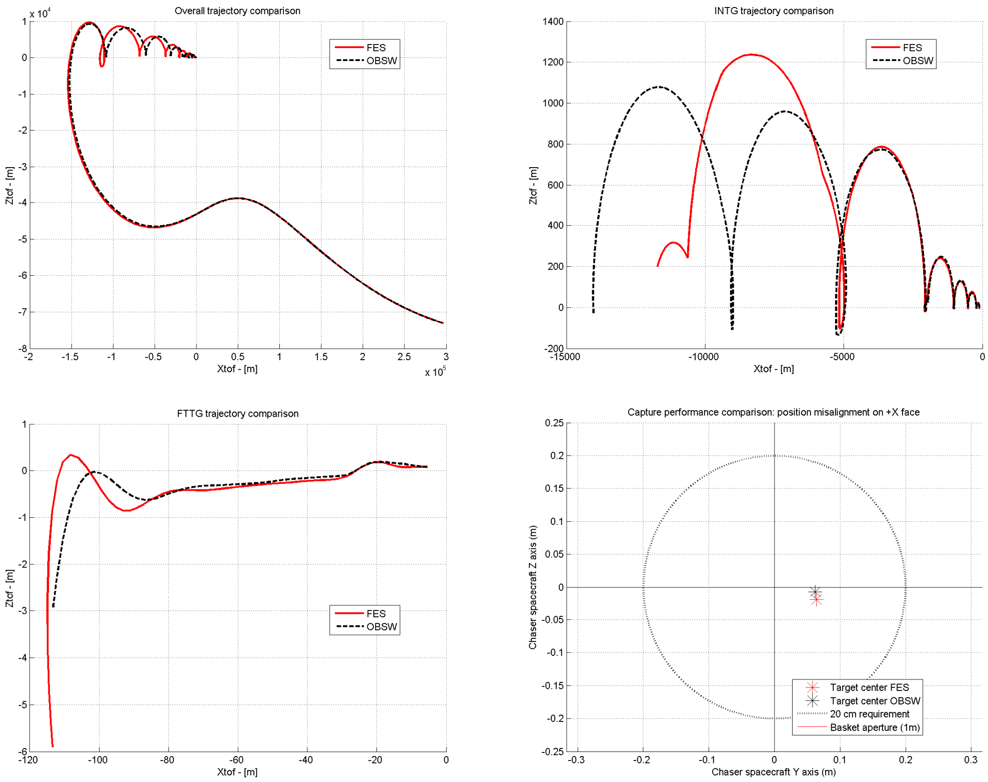

Figure 9 : Test 1: trajectory and final capture accuracy comparison between FES and OBSW

It is possible to appreciate a very good matching between the obtained results. The two trajectories slightly differ at long distances and this is driven by the navigation accuracy. In the design phase but also during the validation campaign the sensitivity of the MPC to this aspect (in particular to the relative velocity estimation accuracy) was identified as the major contributor to the performance. When the distance reduces (INTG, FTTG phases), the navigation improves and the trajectories become almost identical, as well as the final capture accuracy. Finally, also the overall ΔV and propellant consumption are really close in the two simulations (Figure **10**).

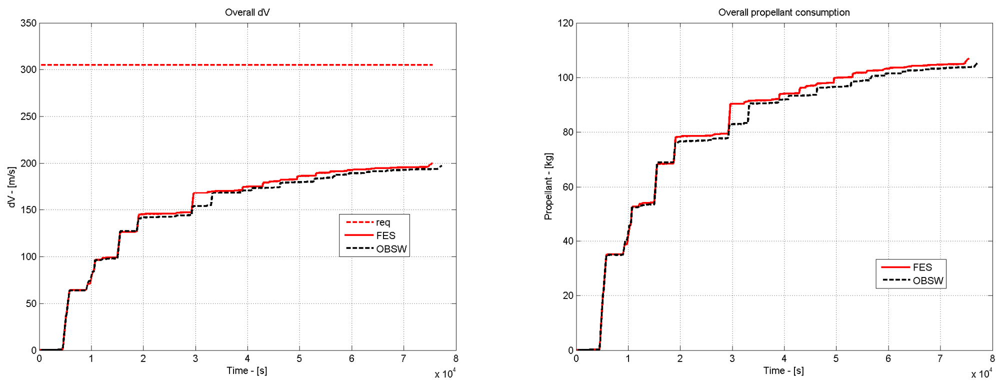

Figure 10 : Test 1: overall ΔV and propellant consumption comparison between FES and OBSW

3.2 Test 2: contingency elliptical orbit

Figure **11** shows the comparison between the trajectories and the final capture accuracy obtained in a “contingency” scenario. The rendezvous orbit is elliptical, but the initial conditions are such that there is a very high drift away from the target, although the initial position is quite close. These conditions allows the detection of the target quite soon, but the guidance is not able to completely stop the drift before the distance is such that the relative sensor (Radio Frequency based) exits from its operational range. This situation causes a period in which the relative navigation degraded in accuracy (propagation) as can be seen in Figure **12**. In both the cases, the Radio Frequency (RF) sensor outage lasts about 4000s.

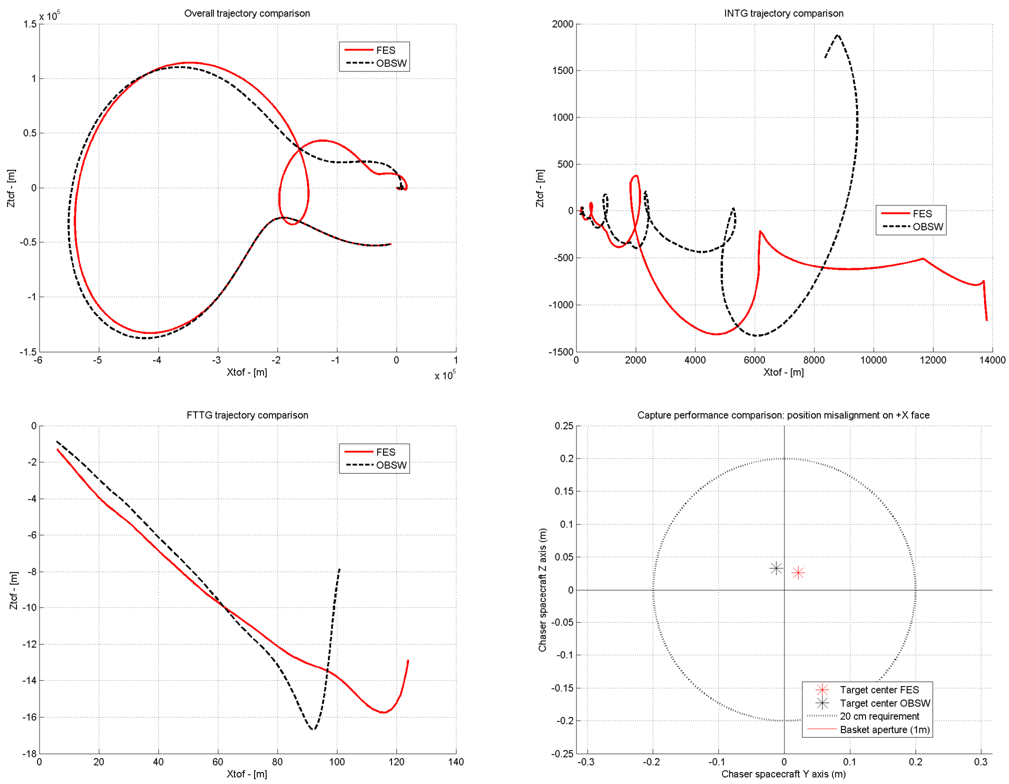

Figure 11 : Test 2: trajectory and final capture accuracy comparison between FES and OBSW

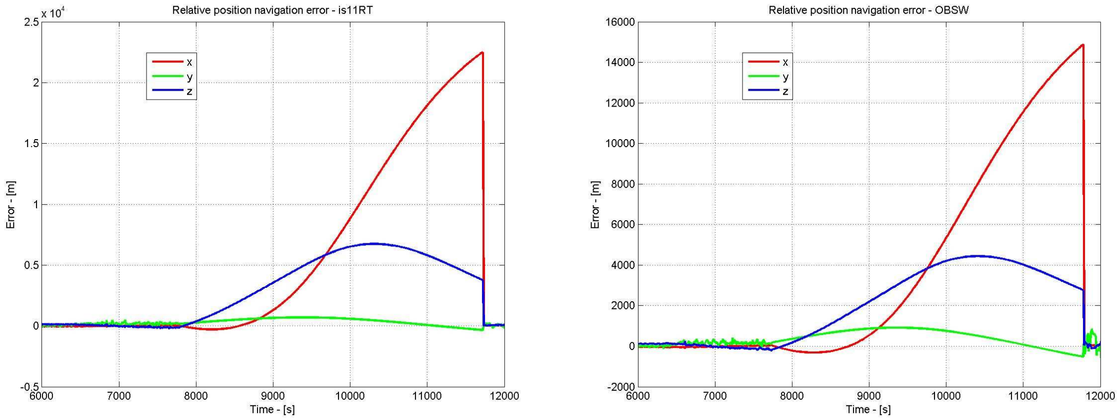

Figure 12 : Test 2: trajectory and final capture accuracy comparison between FES (left) and OBSW (right)

Simulation results show a significant difference in the trajectory at long range (OSTG phase), as a consequence of a different “decision” taken by the MPC when the RF sensors measurements become again available.
Indeed, the degradation of the navigation accuracy slightly differs between FES and Demonstrator, allowing the latter to command a high maneuver to recover the accumulated error immediately after the RF sensor recovery. Instead, the FES performs a higher maneuver later on as can be appreciated in Figure **13**. Exploiting more accurate navigation information in the case of Demonstrator simulation, the MPC is able to perform a better maneuver, which allows both to have a smoother trajectory but also to save a significant amount of propellant (about 30 m/s).

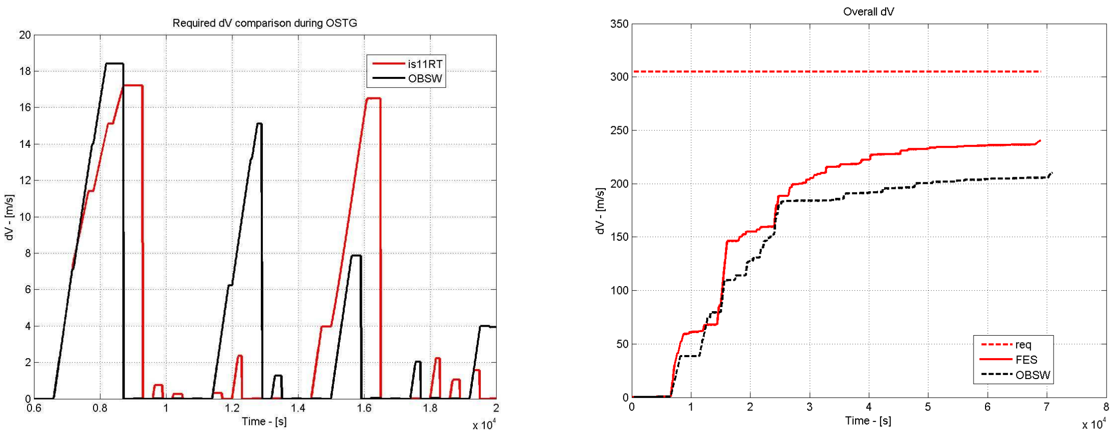

Figure 13 : Test 2: required ΔV (left) and overall ΔV (right) comparison between FES and OBSW

The result of this test case can be considered the synthesis of the ORCSAT project, since it demonstrates the capability of MPC reconfiguration against unexpected events (like a sensor outage) on the basis of the available navigation information.

5. Conclusions

The paper presented the implementation of embedded MPC control system for the rendezvous and capture scenario of Mars Sample&Return mission in a representative flight-like avionic architecture system (Demonstrator). Hardware and software dedicated solutions have been put in place to cope with the peculiarity of the control system, in particular with a processor – co-processor architecture necessary to accommodate optimization algorithms requiring high computational capability. Real-time simulations show that the performance experienced in the validation campaign can be reproduced with the Avionics Demonstrator. Furthermore, the MPC capability of on-line reconfiguration against unexpected events has been further confirmed as a strength point of the designed control system.

References

[1] P. Colmenarejo et al. 2008. HARVD Development, Verification and Validation Approach (from Traditional GNC Design/V&V Framework Simulator to Real-Time Dynamic Testing). 7th International ESA Conference on Guidance, Navigation & Control Systems. Tralee, County Kerry, Ireland.

[2] Strippoli, L et al. 2010. High Integrity Control System For Generic Autonomous RVD. 61st International Astronautical Congress. Prague, CZ.

[3] M. Saponara, V. Barrena, A. Bemporad, E. N. Hartley, J. Maciejowski, A. Richards, A. Tramutola, P. Trodden. 2011. Model Predictive Control application to spacecraft rendezvous in Mars Sample Return scenario. 4th European Conference for Aerospace Sciences (EUCASS). Saint Petersburg, Russia.

[4] M. Saponara, A. Tramutola, P. Creten, J. Hardy, C. Philippe. 2013. Avionic architecture for Model Predictive Control application in Mars Sample&Return rendezvous scenario. Data Systems in Aerospace (DASIA) conference. Porto, Portugal.