SAE TECHNICAL PAPER SERIES

2007-01-2857
E

Design of an Attitude Control System for the Multi-Mission Platform and its Migration to a Real Time Operating System

Marcelo de Lima Bastos Moreira
Equatorial Sistemas S.A.

Marcelo Lopes de Oliveira e Souza
Instituto Nacional de Pesquisas Espaciais

SAE BRASIL
Sociedade de Engenheiros da Mobilidade

FILIADA À
SAE International

Congresso 2007
SAE BRASIL

XVI Congresso e Exposição Internacionais
da Tecnologia da Mobilidade
São Paulo, Brasil
28 a 30 de novembro de 2007

AV. PAULISTA, 2073 - HORSA II - CJ. 1003 - CEP 01311-940 - SÃO PAULO – SP

2007-01-2857

Design of an Attitude Control System for the Multi-Mission Platform and its Migration to a Real Time Operating System

Marcelo de Lima Bastos Moreira
Equatorial Sistemas S.A.

Marcelo Lopes de Oliveira e Souza
National Institute for Space Research

ABSTRACT

This paper presents the first of four parts of the academic design of an Attitude Control System (ACS) for the Multi-Mission Platform (MMP) and its migration to a Real Time Operating System. The MMP is a three axis stabilized artificial satellite now under development at the National Institute for Space Research (INPE). Such design applied some software engineering concepts as: 1)visual modeling; 2)automatic code generation; 3)automatic code migration; 4)soft real time simulation; and 5)hard real time simulation. A block diagram based modeling and a virtual time simulation of the MMP ACS in its nominal operational mode were built in the MatrixX 7.1 environment satisfying the three axis pointing and stabilization requirements. After that, its AutoCode module was used to generate C ANSI code representing the block diagram model. Time characteristics were added to the ACS generated code to make it the real time control software of MMP nominal operational mode. Four operating systems were used for code migration: 1)Windows 2000 (first part); 2)Mandrake Linux 10.1 (second part); 3)RedHawk Linux 2.1 (third part); and 4)RTEMS 4.6.2 (fourth part). Software migration characteristics were described for each of these four computational environments. The results shown the advantages of using those software engineering concepts in the MMP ACS development. They also show excellent agreement between the virtual time simulation results and those obtained with Windows 2000 simulation.

INTRODUCTION

The objective of this work is: 1) analyse, design and simulate a discrete control of MMP nominal operational mode; 2) generate the correspondent real time software; 3) migrate it to the development environment and 4) test it with a real time operating system. The nominal mode is the satellite´s operational mode, where its payload can accoplish its mission. The procedure for development of the software use the concepts of visual modeling, automatic code generation, automatic code migration, soft real time simulation and hard real time simulation.

MULTI MISSION PLATFORM

The Multi Mission Platform (MMP) must be a modular small platform servicing a wide range of scientific applications, communications and low Earth observation. The MMP must be versatile enough to accomplish specific mission need in terms of mass, power, propulsion and data handling capacity.

A vision of MMP, with its solar panel opened and the predicted payload envelop can be seen in Figure 1.

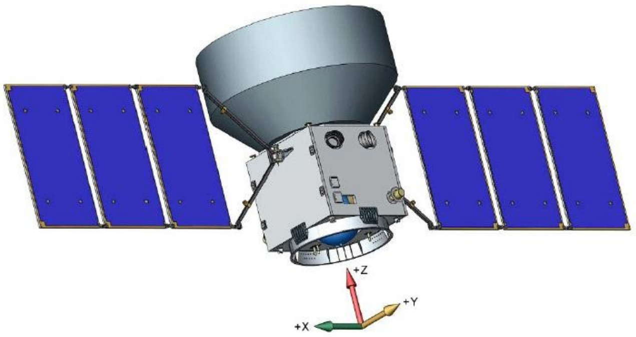

Figure 1 – Multi Mission Platform, with solar painels and payload envelop.

In the nominal operational mode, the mode treated in this work, the satellite attitude must be controlled in three axis to be kept under the following specifications:

Pointing: < 0,05º (3 sigma)

Drift: <0,001º/s

Re pointing of 30º in 180 seconds

MATRIXX

The development of real time systems usually occurs in stages, with separated tools for the project design, software engineering, data acquisition and tests. MatrixX family of products integrates each one of these tools in the system development stage into a single environment. This allows a rapid transition from one stage to the other making it possible to create a functional prototype early in the development process.

Through MatrixX/SystemBuild it is possible to build, simulate, analyse, test and debug a model. MatrixX/AutoCode can be used to generate the source code with real time characteristics in a high level language (C or Ada) for a model. The application generated code can be compiled and migrated for implementation in a wide range of target processors.

The automatic generated code process is part of a concept of fast prototiping, making it possible to create source codes in high level languages from a block diagram model.

The steps for the design, code generation and implementation of the software using MatrixX are described in Figure 2. From these steps, 1 to 3 were performed in MatrixX. Steps 4 and 5 were performed using four computational environments, two of they were composed of real time operating systems.

The present article shows the work done using Windows 2000 computational environment.

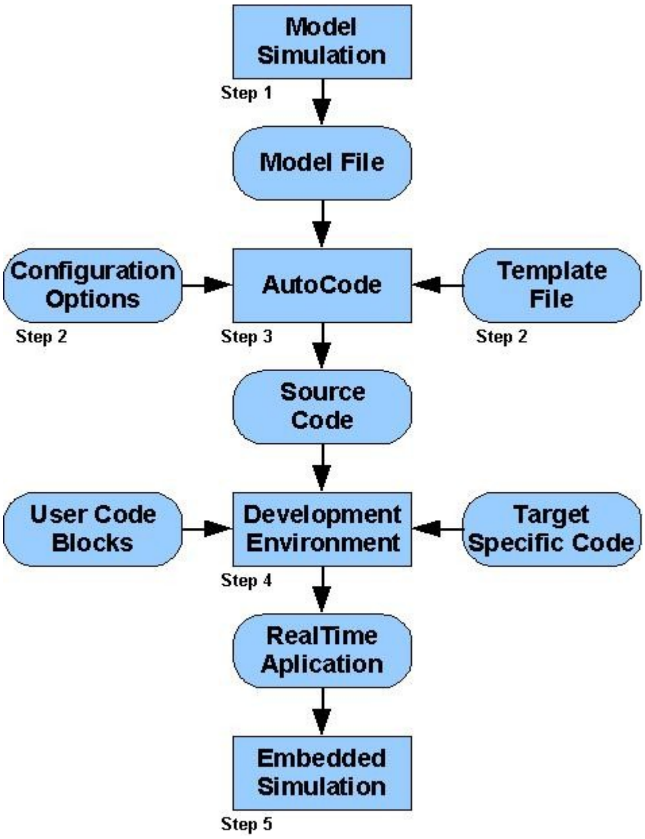

Figure 2 – Process of Modeling, Automatic Code Generation and Implementation.

1. Make the block diagram model using MatrixX/SystemBuild and simulate it until its performance are satistactory.

2. Configure the model of the generated code using a default template. Steps 1 and 2 are the inputs to the automatic code generation.

3. Automatic code generation to C or Ada language. In this project the choosen language was C, because of the availability of C compilers for the operational system used in this work.

4. Compilation of the generated code with the specific codes of the target processor and operational system.

5. Simulation of the real time application must be performed in the choosen operational system to compare the results with the ones generated using the model developed in the step 1.

MODELING

The model of the Multi Mission Platform follows the equations of motion of a rigid spacecraft [5], considering the presence of a reaction wheel and its coupler with the satellite [4]. The plant equation is:

$$I\dot{\vec{\omega}} + \vec{\omega} \times I\vec{\omega} = \vec{\omega} \times h - \dot{h} + T_P \qquad (1)$$

where I is the inertial matrix of the satellite, $\vec{\omega}$ is the angular velocity vector of the satellite, h is the angular moment of the reaction wheels, *dh/dt* is the variation of the angular moment of the reaction wheels, that represents the controllable part of equation (1), also called control torque and T_P is the external perturbation torque.

The differential kinematic equations of a rigid body in orbit can be seen in some work [5] and [1], it is described in equation (2).

$$\begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta & sin\varphi\, sin\theta & \cos\varphi\ \cos\theta \\ 0 & \cos\varphi\ \cos\theta & -\ sin\varphi\ \cos\theta \\ 0 & sin\varphi & \cos\theta \end{bmatrix} \begin{bmatrix} \vec{\omega}_1 \\ \vec{\omega}_2 \\ \vec{\omega}_3 \end{bmatrix} - \qquad (2)$$
$$+\ \omega_0 \begin{bmatrix} sin\theta \\ \cos\theta\ \cos\phi \\ sin\theta\, ssin\phi \end{bmatrix}$$

where *roll*, *pitch* e *yaw* (φ, θ, ϕ) are the Euler rotation angles, described in [1], [2], [4] and [5].

SIMULATIONS

For the simulations MatrixX/SystemBuild was used, making it possible to develop the project in a visual form, the block diagrams. Block diagrams shows the project in a system point of view, and the same time allows a detailed visualization of its internal sub-blocks. In that way it's easy to change from one point of view to the other.

A block diagram of the system can be seen in Figure 4. The control law used was Proportional, Integral and Derivative (PID) in relation to the attitude error of the plant.

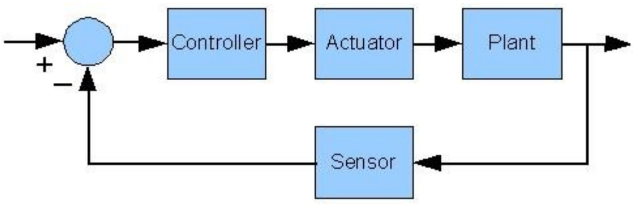

Figura 4 – Closed-loop representing the control system of MMP satellite.

The objective of the simulations performed was to demonstrate the project requirements acomplishment.

The simulations were performed with initial condition of (30, 0, 0) degrees in attitude error and angular velocity of (0, -0.062, 0) °/second.

RESULTS

Figure 5 represents the comparison between the model block diagram simulated and the code migrated to Windows 2000 of one of the systems variables.

It can be noted that the results are compatible. With this comparison the generated code can be considered the correspondence of block diagrams model developed in MatrixX/SystemBuild.

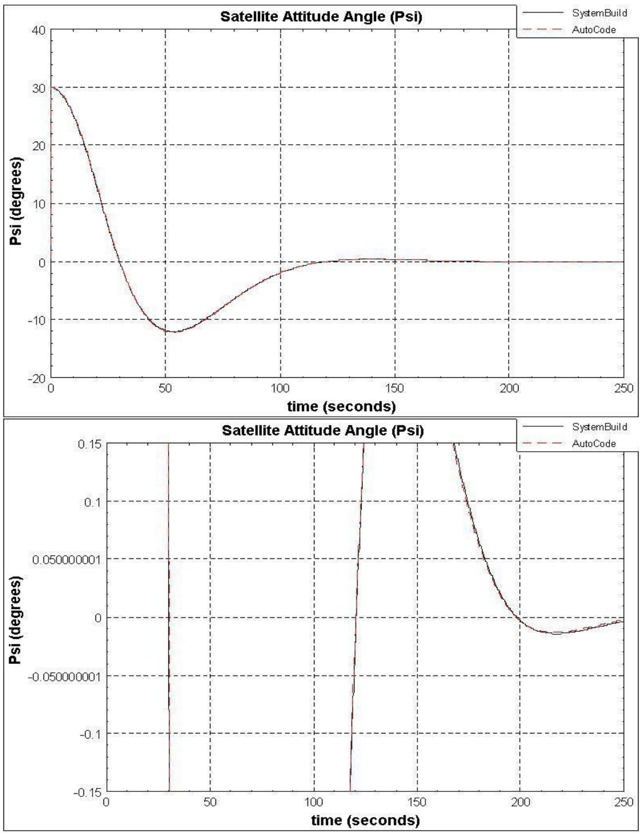

Figure 5 – Comparison of the attitude angle (Psi) simulated and migrated to Windows 2000.

CONCLUSION

The system state variable behavior graph shows the compatibility of the results obtained in the simulation and in the code generation with its migration to an operating system.

The automatic code generation is capable not only of obtain accurate logic results but also in reduce the time necessary to migrate a code to an operating system.

BIBLIOGRAPH

[1] Arantes Jr., G. Estudo Comparativo de Técnicas de Controle de Atitude em 3 Eixos para Satélites Artificiais. 187p. Dissertação (Mestrado em Ciência Espacial) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2005.

[2] Kaplan, H.M. Modern Spacecraf Dynamics and Control. 1st. ed. New York – USA, John Wiley & Sons, Inc., 1976, 415p.

[3] Ogata, K. Modern Spacecraft Dynamics and Control. 3rd ed. New Jersey – USA, Prentice-Hall, 1997. 997p.

[4] Wertz, J. R. Spacecraft Attitude Determination and Control. 1st ed. Dordrecht, Holland: D. Reidel Publishing Co., 1978. 858p.

[5] Wie, B. Space Vehicle Dynamics and Control. 1st ed. Reston, Virginia, USA. AIAA Educational Series. 1998. 661p.

ISSN 0148-7191