ISSN 2176-5480

22nd International Congress of Mechanical Engineering (COBEM 2013)
November 3-7, 2013, Ribeirão Preto, SP, Brazil

A HARDWARE-IN-THE-LOOP SIMULATION APPROACH FOR TESTING CONTROL SYSTEMS OF SMALL SATELLITES

Leandro de Oliveira Rodrigues
Instituto Tecnolgico de Aeronáutica, Departamento de Engenharia Eletrônica e Computação,Pç Mar. Eduardo Gomes, 50 - Vila das Acácias - CEP 12228-900 - São José dos Campos - São Paulo
leandro.olirodrigues@gmail.com

Marsil de Athayde Costa e Silva
Instituto Tecnológico de Aeronáutica, Departamento de Engenharia Aeronáutica e Mecânica, Pç Mar. Eduardo Gomes, 50 - Vila das Acácias - CEP 12228-900 - São José dos Campos - São Paulo
marsil@ymail.com

Rafael Barbosa Januzi
Universidade Federal de São Paulo, Instituto de Ciência e Tecnologia,R. Talim, 330 - Jardim Aeroporto - CEP 12231-280 - São José dos Campos - São Paulo
rjanuzi@gmail.com

Neusa Maria Franco de Oliveira
Instituto Tecnológico de Aeronáutica, Departamento de Engenharia Eletrônica e Computação, Pç Mar. Eduardo Gomes, 50 - Vila das Acácias - CEP 12228-900 - São José dos Campos - São Paulo
neusa.m.franco@gmail.com

Emilia Villani
Instituto Tecnológico de Aeronáutica, Departamento de Engenharia Aeronáutica e Mecânica, Pç Mar. Eduardo Gomes, 50 - Vila das Acácias - CEP 12228-900 - São José dos Campos - São Paulo
evillani@ita.br

Abstract. *The procedure of testing devices for artificial satellites needs great attention because once the satellite is launched, it is almost impossible to correct some defective device. Thus, this task should be very carefully designed and executed. However, the environment for tests is limited to a simulation environment and in the best case a complete prototype should be built. So, the use of hardware-in-the-loop (HIL) techniques is a very promising approach which may increase the reliability of the tests. This approach uses both simulated and real devices in a single loop in order to evaluate the performance of the real portion which is working as it will be in the completely real plant. It allows considerations such as computing limitations and analog-digital conversion's resolution. This paper proposes an implementation of a HIL system which aims to validate the electronic acquisition modules of sensors signals, the control driver of actuators and the software responsible for the attitude control of small satellites. This methodology includes a micro-controller in the loop executing the control routine while the environment is simulated in a computer. The results demonstrate the effectiveness of this implementation and its methodology against the use of a completely simulated environment.*

Keywords: *hardware-in-the-loop simulation, small satellites, attitude control, embedded systems*

1. INTRODUCTION

Most artificial satellites need an attitude control system to keep it into a desired condition of flight; the condition, which is dependent of the mission, may be pointing a device to some direction, e.g. pointing antennas to Earth or solar panels to Sun, or protecting it from Sun light, e.g. cameras or some face without solar cells. Thus the role of such an attitude control system may be of high importance since the mission might fail if the attitude is not maintained within the acceptable limits.

Thus, the design of attitude control systems for satellites must be very carefully conducted in order to eliminate possible failures. In this context hardware-in-the-loop (HIL) simulations has demonstrated to be a very promising approach in aiding the design of such systems (Jung *et al.*, 2013; Park *et al.*, 2013). This approach considers the inclusion of some real device in the simulation loop; the device and all its designed features work as they will in the real conditions but within a simulated environment. This improves tests in terms of reliability and decrease the development time.

According to Leitner (1996) a HIL simulation makes part of the second phase in a space technologies development project. After built the subsystem's equipment (first phase), the next step will be the interfacing of the payload or subsystem with the simulated spacecraft in a hardware-in-the-loop sense. It has several benefits within the project, including

ISSN 2176-5480

showing some integration problems before put a system into space, exploring scenarios for in-flight anomalies and hardware and software failures, evaluating various potential mission concepts and others.

This paper proposes a HIL platform to test control algorithms for a small satellite. The platform contains a microcontroller, playing the role of the on board computer, and an analog-digital converter; both devices are the hardware in the simulation loop. A simple control algorithm (Carrara *et al.*, 1992) is implemented in the microcontroller with the purpose of testing the platform. The approach presented in this paper can be easily applied to the design of a control system for other small satellites in addition to the studied case; the modularity of the system facilitates the process of porting the platform to other case with the only changes to be done in the orbit parameters, in the control algorithm which depends on the requiriments of the mission, and the on board computer used. Results demonstrate the effectiveness of the constructed platform.

The paper is organized as follows: section 2 presents the HIL implemented, section 3 explains the control algorithm used in tests, section 4 shows the setup of the simulation, section 5 presents the results, and finally section 6 concludes the paper.

2. HARDWARE-IN-THE-LOOP IMPLEMENTATION

The simulation environment is composed of basically two main components; the first is the satellite's dynamical model which is simulated in a single computer using STK software and MATLAB, and the second is the emulation of sensors' signals together with the on-board computer. Figure 1 shows the diagram of the control loop.

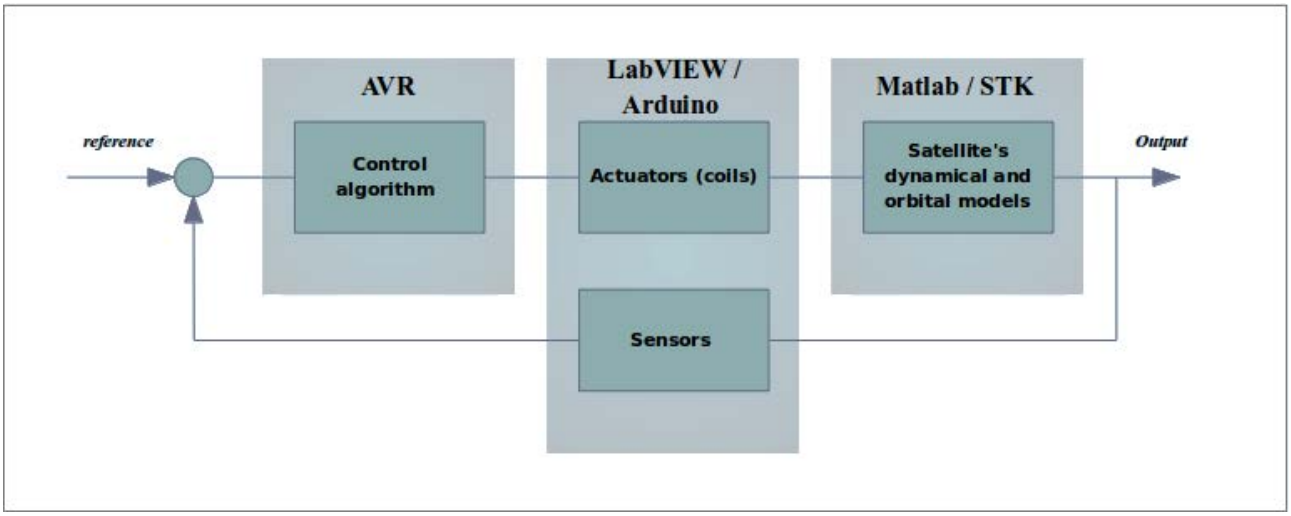

Figure 1. Control loop.

STK is responsible for the orbit propagation, environment signals supply, and also shows the dynamic motion of the satellite through a 3D model. MATLAB is responsible for controlling all communication interface with STK and LabVIEW, and also calculates the values of the sensors signals and integrates the satellite's attitude dynamic model. To emulate the signals from these sensors LabVIEW is used, which receives data from MATLAB and controls a hardware that generates electrical signal referring to the values of the satellite sensors. This hardware uses an Arduino and an electric circuit that condition and digitalizes the signals. After this stage, the data are sent to microcontroller AVR-32, that represents the onboard computer of the satellite. Within AVR-32 runs a control algorithm that uses sensors data to perform its calculations. So, AVR-32 sends to MATLAB, the commands to drive the satellite, which are used in the integration of dynamic model. This feeds back the system that now, has a new attitude and new parameters of sensor readings, repeating the cycle every integration step. Next subsections present more detailed informations.

2.1 MATLAB and STK environment

This group is responsible for the simulation environment of the HIL Platform. STK has the function to simulate the satellite attitude enabling the view of its orbit's evolution through the 3D model. But its main function is to provide the environment parameters such as the Sun's direction vector, insolation on solar panels data that allow calculating the values of these sensors, and to generate the reading of Earth's magnetic field that represents the values of magnetic sensor (magnetometer). A custom report was created in STK to provide the desired values and during the simulation running a report of these data is requested every iteration.

MATLAB is the responsible to synchronize all communications in this testing platform. It has communication with STK through TCP/IP communication protocol using the connect library provided by STK, and also communicates with

ISSN 2176-5480

the LabVIEW via TCP/IP too.

In the MATLAB's main function the dynamic model (equation (1)) of the satellite is integrated and sensors signals are calculated based on the values provided by STK (Sun's direction and geomagnetic field). To do this, MATLAB needs getting data from the report created by STK. After that, it sends this data package to LabVIEW that controls and configures the signals and send them to Arduino.

The dynamical model of the satellite is presented by (1) (Chobotov, 1991; Wertz, 1978) where T_x, T_y, and T_z are the external torque components, w_x, w_y, and w_z are the angular rate components and I_x, I_y, and I_z are the diagonal elements of the inertia matrix.

$$\begin{cases} \dot{w}_x = \frac{(I_z - I_y)}{I_x} w_y w_z + T_x \\ \dot{w}_y = \frac{(I_z - I_x)}{I_y} w_x w_z + T_y \\ \dot{w}_z = \frac{(I_x - I_y)}{I_z} w_x w_y + T_z \end{cases} \tag{1}$$

Moreover, MATLAB receives control data from AVR-32, through LabVIEW. These values will be used in the dynamic model originating a new attitude quaternion which varies according to the activated actuator. So, MATLAB must set the new attitude in the STK in each time step.

2.2 Hardware configuration

In order to emulate the sensor signals, using voltage outputs, a signal generation board was assembled. This hardware is controlled by LabVIEW and uses an Arduino as the signal provider.

The main program developed in LabVIEW communicates with the Arduino via a USB port, having a serial port function.

The Arduino PWM output pins have been used to originate the sensors signals, such as magnetometer and solar panels. These signals have to pass through a RC filter in order to transform them into an analog reference signals. By the fact that PWM signals ranges between 0 and 5 volts, there is the need to make equivalence with the values range of the sensors. To do this, it is necessary controlling the duty cycle of the PWM signal. In this HIL Platform, the angular rates signals are not emulated by Arduino, they pass directly from LabVIEW to AVR-32 using the USART communication protocol. It is done because the Attitude Control Subsystem haven't defined the gyros yet, neither its communication interface.

These signals have to be sent to AVR-32 microcontroller. So, there is a stage where the data pass through Analog Digital Converter (ADC). The communication interface used between ADC and AVR-32 is SPI (Serial Peripheral Interface).

After get the data from ADC, the AVR-32 runs its control function. Thus, the microcontroller sends the command to drive the actuators to MATLAB, using its communication bridge with LabVIEW.

An important point in this Platform is that AVR-32 sends to LabVIEW the data about the sensors signals read by microcontroller from ADC. This way turns possible a comparison between the data obtained from software and the data emulated by the hardware.

2.3 AVR software

The software that runs into the AVR-32 processor is a sequential program, with an infinite loop main code. Below is presented a flowchart with the program description.

The program is split in some smaller activities, each activity is described next.

2.3.1 Initialize AVR32

In this part of program all configurations of the AVR-32 processor are done; the necessary configurations are: 1) select the main clock for the processor, in this case is used a clock of 16 MHz. 2) Initialization of all pins of communication, like GPIOs (General Purpose Input/Output), SPI (Serial Peripheral Interface Bus) and USART (Universal Asynchronous Receiver/Transmitter). 3) Initialization of the LCD (Liquid-Crystal Display) screen of the Kit of AVR-32 to permit real-time monitoring of the running algorithm.

2.3.2 Wait 5 seconds

This delay-time after the initialization of AVR-32, exists to give a time to all other equipments in the experiment be initialized to, for example the analogical-digital converter, who receives some signals of the AVR-32 to be configured. The delay time is also used for others equipments in the experiment have a settling time.

2.3.3 Receive data from Matlab

This part of program is the beginning of the infinite loop of the algorithm, here the AVR-32 receives the angular rate data. This data is generated in Matlab and transmitted to AVR-32 by the LabVIEW Software, the communication protocol

ISSN 2176-5480

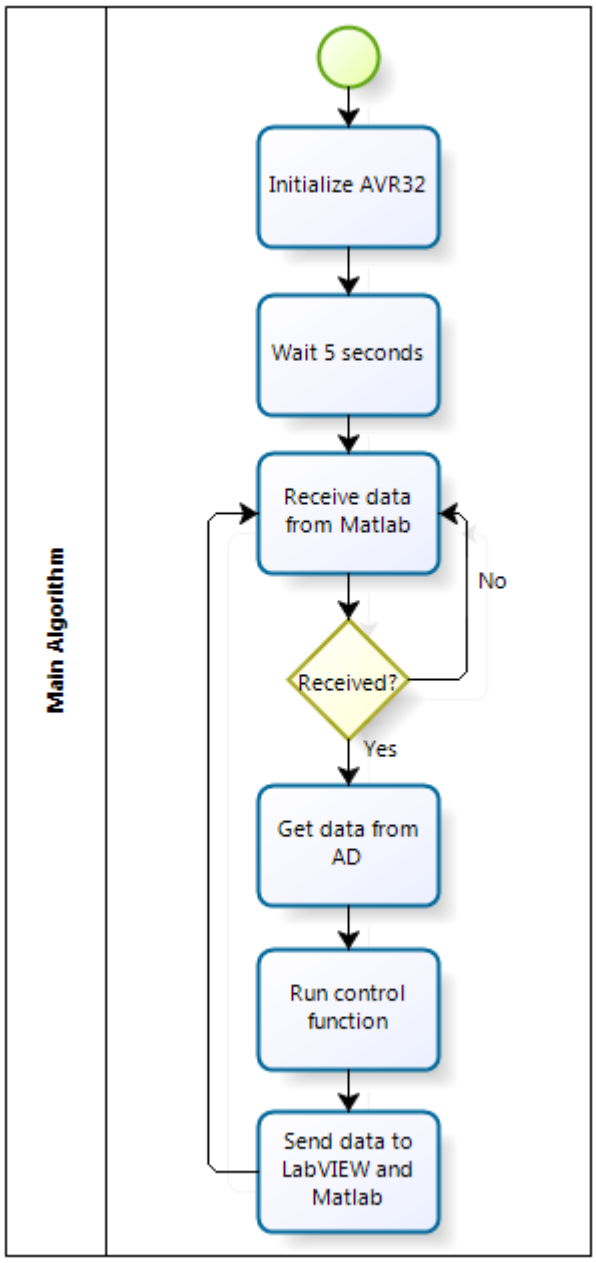

Figure 2. Main algorithm of AVR-32 processor.

used is the USART protocol. The program waits at this point until the data are received.

2.3.4 Get data from AD

In this part of the algorithm, the program gets the data of analogical-digital converter, the data are: 3 magnetometers's data (axes x,y and z) and 5 solar sensors's data. The AVR-32 have to send a signal to the analogical-digital converter initiate the conversions e generate the requested data, this signal is sent through a GPIO pin, in sequence the AVR-32 reads all data through a SPI communication.

2.3.5 Run control function

Here the control function will be executed, the function will use all the data acquired from the others equipments in the experiment, these data are: angular rate from the Matlab, magnetometers's data and solar sensors's data from analogical-digital converter. The control function will generate the information of what coils will be activated.

ISSN 2176-5480

2.3.6 Send data of AD to Matlab

At last the program will send the data acquired of analogical-digital converter and the data generated by the control function, to LabVIEW and to Matlab. The LabVIEW will get the analogical-digital data and save it in a file, and the Matlab will get the coils's information generated by the control function. After this process the program will reinitiate the infinite loop.

3. CONTROL SCHEME

The case study considered in this paper is a small satellite with approximately $38.79kg$ and size 60x60x52cm. The satellite has three coils as the actuator one in each axis; the coil placed in Z axis is responsible for the pointing of its spin axis. The other coils are responsible for the spin gain and are redundant, i.e. if one doesn't work properly then the other is used.

The satellite is injected in orbit with an initial condition and it should achieve a desired spin rate about its maximum inertia moment axis. This is done using an electromagnetic coil with $5Am^2$ of magnetic dipole moment. The satellite is spinned to provide greater stability.

The coil interacts with the geomagnetic field and produces torque according to the following equations:

$$\vec{T} = \vec{M} \times \vec{B} \tag{2}$$

where $\vec{T}$ is the produced torque, $\vec{M}$ is the magnetic dipole moment, and B is the geomagnetic field measured by AVR.

The control algorithm uses the polarity of the coil as its control variable and there are two constraints limiting the actions of the controller: only one action is made within a single iteration of the algorithm since it is a discrete controller and the magnetic dipole moment of the coil is constant. These two constraints may decrease the efficiency of the controller, however, as the goal is just to spin the spacecraft, there are no great losses.

The coil is placed in the Y axis and the Z is the spin axis. Thus the controller considers the geomagnetic field measured along the X axis because the cross product between B_x and M_y will generate a torque in Z. This way the polarity P of the coil is calculated using the following equation where B_x is the X component of B:

$$P = sign(B_x) \tag{3}$$

In order to avoid issues caused by uncertainty in measures, the coil is turned off when the field is crossing zero and also different limits are used to turn it on and off. Figure 3 shows an example for this case where the limit to turn it on is 15% and 10% to turn it off.

In each iteration of the control algorithm, the polarity is chosen accordingly to the above mentioned rules and it remains the same until the next iteration.

4. SIMULATION SETUP

Due the fact the user can choose your desire scenario simulation, the simulations parameteres can be changed as needed.

The simulation setup, in this case study, considered a satellite with a Low Earth Orbit (LEO) about $600km$ and an inclination of 87.3° (polar orbit), both them must be setup in STK and MATLAB. The orbital propagator chosen was the Two-Body, because of a need to create a simple test scenario to analysis the attitude behavior. This propagator must be chosen in STK's configurations. The initial angular rates choose was as follows: $w_x = 3rpm$, $w_y = 3rpm$, and $w_z = 2rpm$.

The inertia matrix was:

$$I = \begin{bmatrix} 2.56 & 0 & 0 \\ 0 & 2.56 & 0 \\ 0 & 0 & 2.97 \end{bmatrix} \tag{4}$$

The time step was set at 1 second and the simulation time (end time of the simulation) was configured to take 5 orbits. This configuration must be set at MATLAB. The orbital period was chosen to start on January 25, 2013 and to finish on February 25, 2013, so, this ensures the simulation time.

5. RESULTS

A complete simulation was run using the HIL platform and the goal was to spin the spacecraft about its Z axis until 12 revolutions per minute (rpm). Figure 4 shows the angular rates after 5 orbits. It is possible to see that the desired rate is achieved after 2.5 orbits. Figure 5 shows the nutation angle after 5 orbits; the realtively small value after 2.5 orbits

ISSN 2176-5480

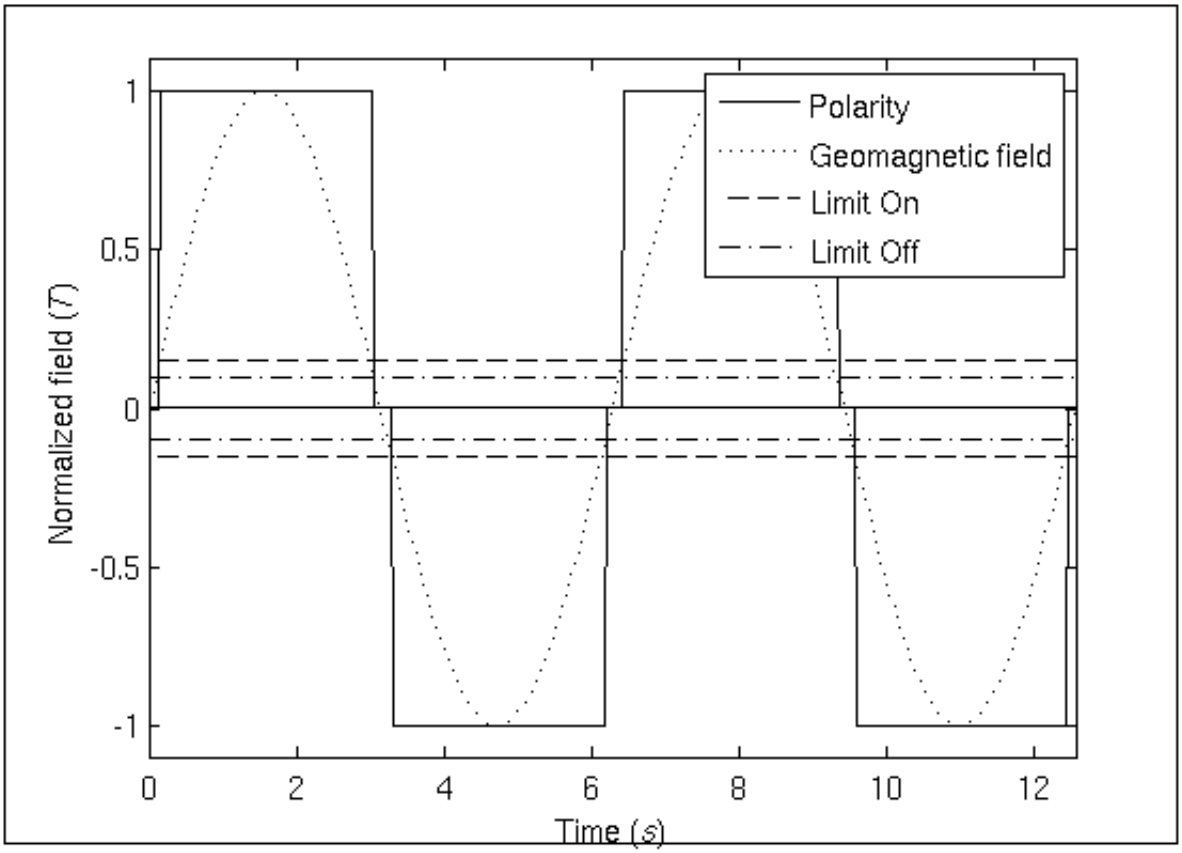

Figure 3. Coil polarity for controller.

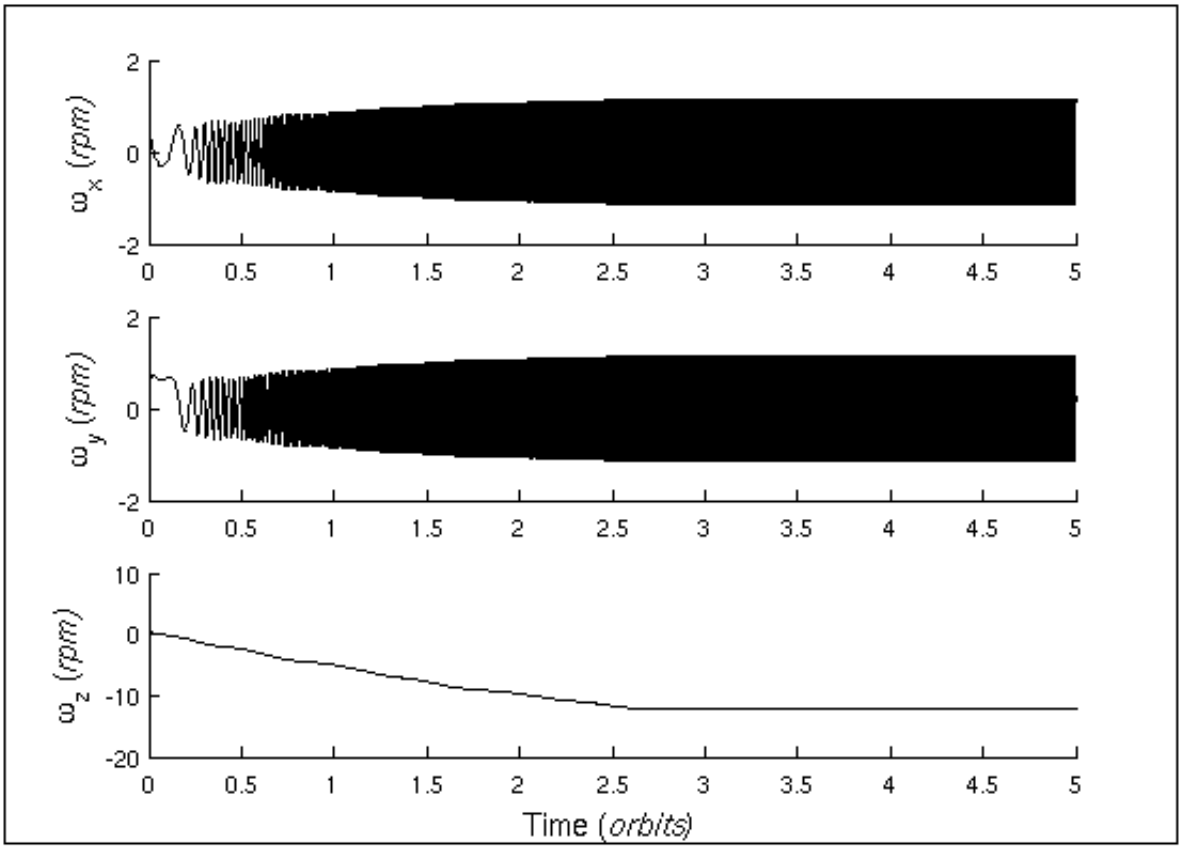

Figure 4. Angular rates of spacecraft.

ISSN 2176-5480

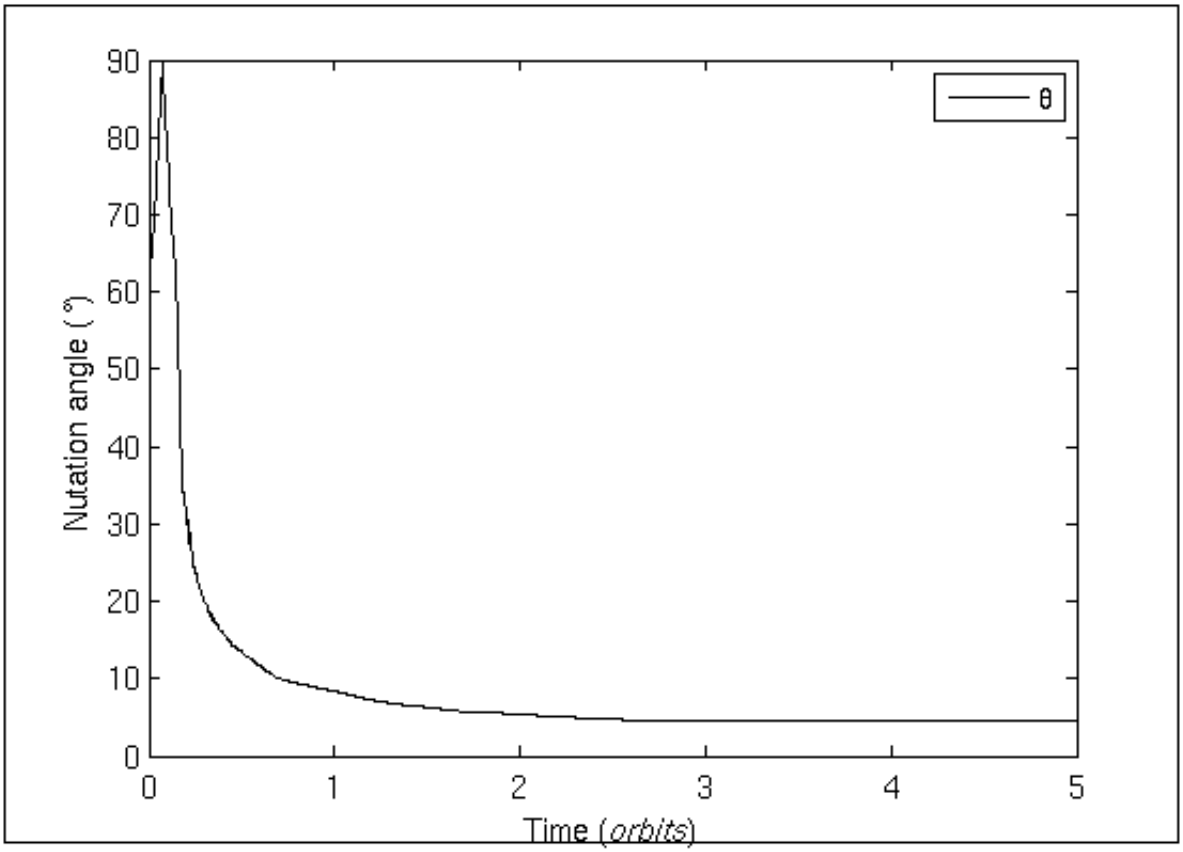

Figure 5. Nutation angle.

indicate that the spin axis is very close to the satellite's Z axis. This behaviour is the expected one since the major inertia is in that direction. The geomagnetic field measured by AVR (right) is compared against the true values (left) in figure 6. Also, table 1 presents the correlation between the measured and the true values; values in parenthesis are the p-values for the respective correlation. The first value in each cell is the correlation and the second is the p-value for the correlation.

Table 1. Correlation between magnetic field provided by STK (B) and measured by AVR ($\bar{B}$).

	B_x	$\bar{B}_x$	B_y	$\bar{B}_y$	B_z	$\bar{B}_z$
B_x	1.0000 (1.0000)	0.9010 (0.0000)	0.0045 (0.4404)	-0.5559 (0.0000)	-0.0114 (0.0511)	-0.0128 (0.0294)
$\bar{B}_x$	0.9010 (0.0000)	1.0000 (1.0000)	0.3184 (0.0000)	-0.2669 (0.0000)	-0.0087 (0.1372)	-0.0100 (0.0893)
B_y	0.0045 (0.4404)	0.3184 (0.0000)	1.0000 (1.0000)	0.7432 (0.0000)	-0.0051 (0.3798)	-0.0029 (0.6242)
$\bar{B}_y$	-0.5559 (0.0000)	-0.2669 (0.0000)	0.7432 (0.0000)	1.0000 (1.0000)	-0.0074 (0.2093)	-0.0052 (0.3733)
B_z	-0.0114 (0.0511)	-0.0087 (0.1372)	-0.0051 (0.3798)	-0.0074 (0.2093)	1.0000 (1.0000)	0.9947 (0.0000)
$\bar{B}_z$	-0.0128 (0.0294)	-0.0100 (0.0893)	-0.0029 (0.6242)	-0.0052 (0.3733)	0.9947 (0.0000)	1.0000 (1.0000)

The Sun angles are presented in figure 7. The values in left are the true values and the measured values are in the right. Table 2 shows the correlation between the measured and the true values of Sun direction. Note that when the spacecraft enters in the eclipse zone the values measured are all zeros which is interpreted as if the Sun were perpendicular to the normal of the panels. The direction in the Z axis also has this behaviour when the Sun is lightning the Z^- face of the satellite which has no solar panels.

ISSN 2176-5480

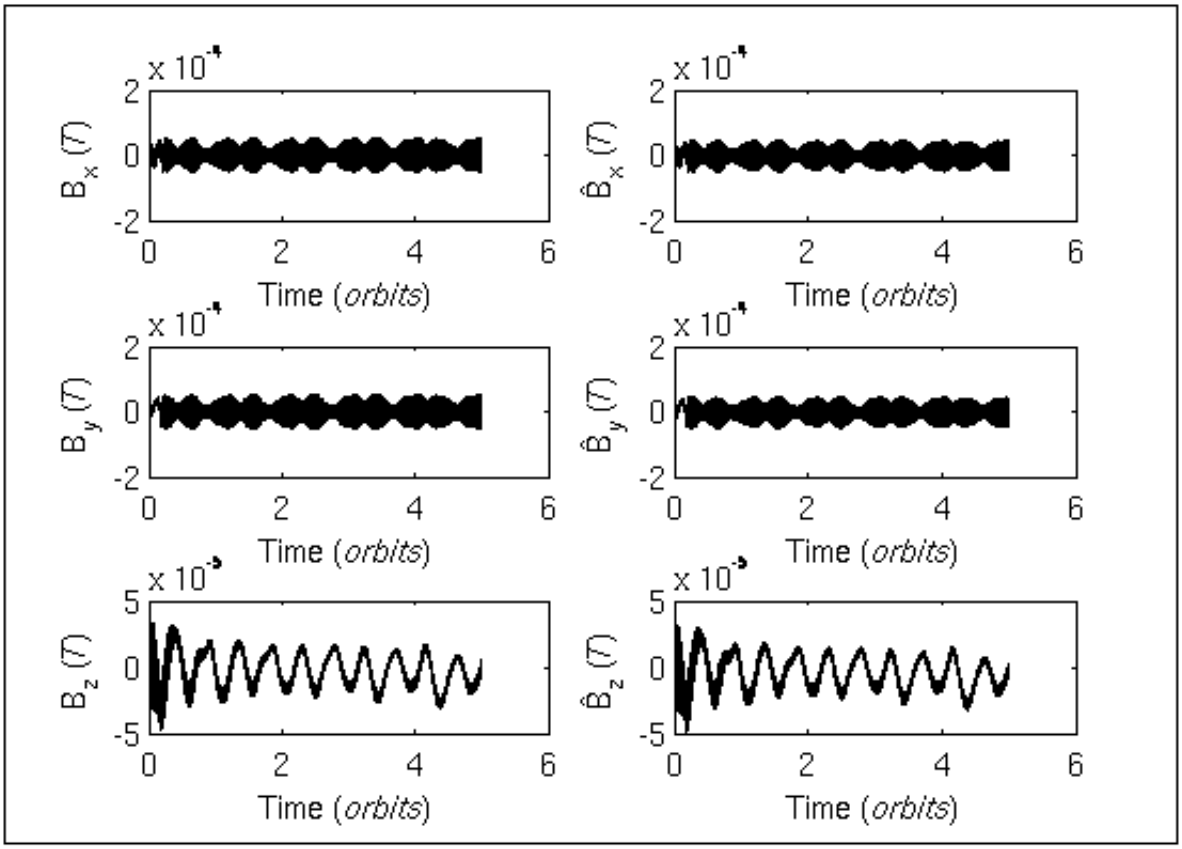

Figure 6. Magnetic field provided by STK (B) and measured by AVR ($\hat{B}$).

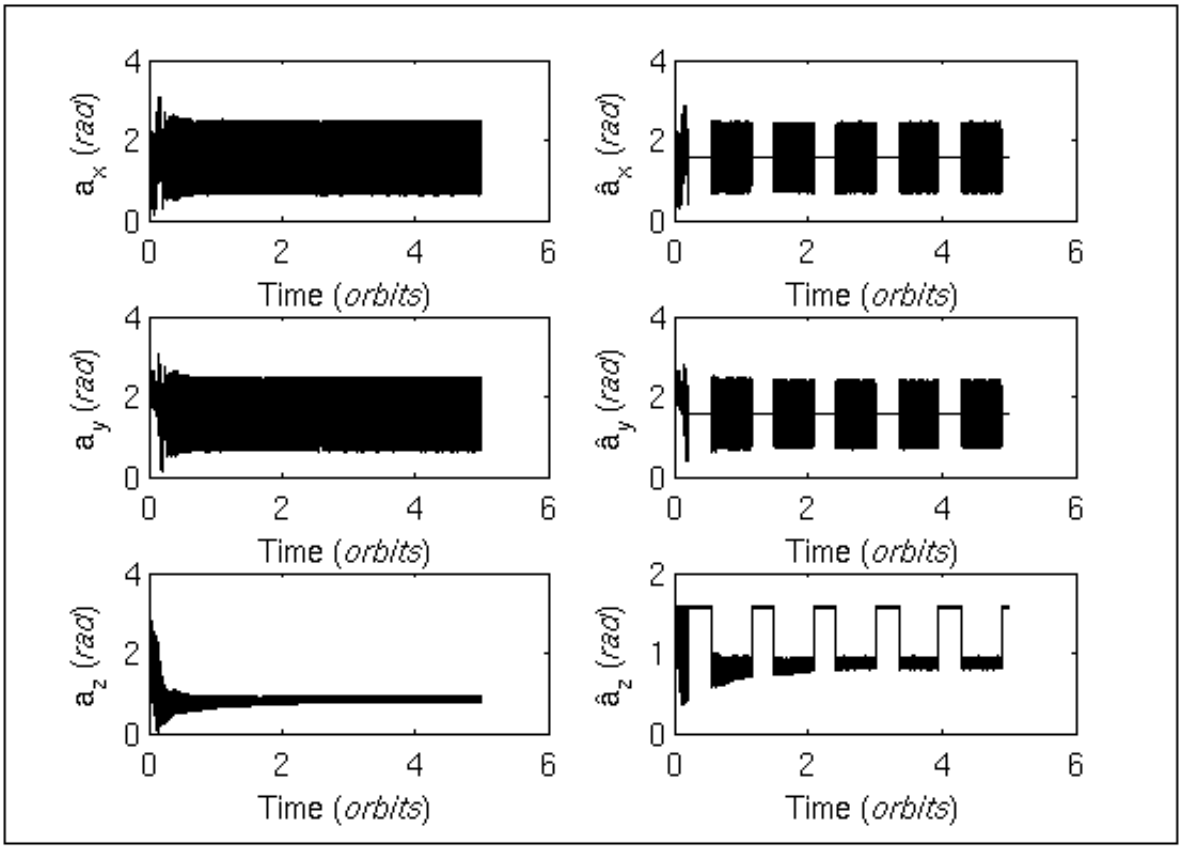

Figure 7. Sun angle provided by STK (a) and measured by AVR ($\hat{a}$).

ISSN 2176-5480

Table 2. Correlation between the Sun direction provided by STK (S) and measured by AVR ($\bar{S}$).

	S_x	$\bar{S}_x$	S_y	$\bar{S}_y$	S_z	$\bar{S}_z$
S_x	1.0000 (1.0000)	0.8063 (0.0000)	0.0001 (0.9869)	-0.0389 (0.0000)	0.0115 (0.0500)	0.0051 (0.3867)
$\bar{S}_x$	0.8063 (0.0000)	1.0000 (1.0000)	0.0167 (0.0044)	-0.0275 (0.0000)	0.0140 (0.0167)	0.0069 (0.2361)
S_y	0.0001 (0.9869)	0.0167 (0.0044)	1.0000 (1.0000)	0.8035 (0.0000)	0.0990 (0.0000)	0.0182 (0.0019)
$\bar{S}_y$	-0.0389 (0.0000)	-0.0275 (0.0000)	0.8035 (0.0000)	1.0000 (1.0000)	0.1230 (0.0000)	0.0212 (0.0003)
S_z	0.0115 (0.0500)	0.0140 (0.0167)	0.0990 (0.0000)	0.1230 (0.0000)	1.0000 (1.0000)	0.2101 (0.0000)
$\bar{S}_z$	0.0051 (0.3867)	0.0069 (0.2361)	0.0182 (0.0019)	0.0212 (0.0003)	0.2101 (0.0000)	1.0000 (1.0000)

6. CONCLUSIONS

The constructed platform was able to simulate a control algorithm for the small satellite studied. The sensors' signals were correctly emulated and read by the on-board computer (AVR) and the correlation between the true and measured signals was significantly considering 5% of significance.

The control algorithm implemented was able to spin the satellite until the desired rate. The discretization of the control scheme did not bring any additional issues, however if a greater speed were desired, problems with the sampling time would appear.

Each iteration of the simulation lasts about one second, thus the time consumed to simulate several orbits is fairly long which may prevent to consider, for example, simulation of a year of mission.

Future works will be testing a pointing control algorithm to improve the power generation or to correctly point an antenna.

7. REFERENCES

Carrara, V., Padilha, O.S., Varotto, S.E.C. and Ricci, M.C., 1992. "Um experimento de teste da bobina de rotacao do scd2". RPQ INPE-5404-RPQ/658, INPE, Sao Jose dos Campos.

Chobotov, V.A., 1991. *Spacecraft Attitude Dynamics and Control*. Orbit Book CO.

Jung, J., Park, S.Y., Kim, S.W., Eun, Y.H. and Chang, Y.K., 2013. "Hardware-in-the-loop simulations of spacecraft attitude synchronization using the state-dependent riccati equation technique". *Advances in Space Research*, Vol. 51, No. 3, pp. 434 – 449.

Leitner, J., 1996. "Space technology transition using hardware in the loop simulation". In *Aerospace Applications Conference, 1996. Proceedings., 1996 IEEE*. Vol. 2, pp. 303–311 vol.2.

Park, H.E., Park, S.Y., Park, C. and Kim, S.W., 2013. "Development of integrated orbit and attitude software-in-the-loop simulator for satellite formation flying". *Journal of Astronomy and Space Science*, Vol. 30, No. 1, pp. 1–10.

Wertz, J.R., 1978. *Spacecraft attitude determination and control*. D. Reidel Publishing Company.

8. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.