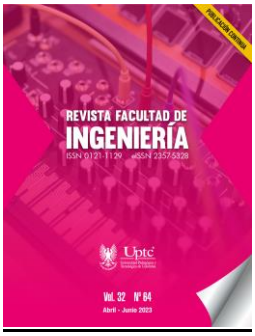
Revista Facultad de Ingeniería

Journal Homepage: https://revistas.uptc.edu.co/index.php/ingenieria



Software Integration of an IMU Sensor to a CubeSat Platform Based on CSP (CubeSat Space Protocol)

Sergio-Fernando Barrera-Molano[1]

Javier-Enrique Méndez-Gómez[2]

Dib-Ziyari Salek-Chaves[3]

Received: March 13, 2023 **Accepted:** June 21, 2023 **Published:** June 30, 2023

Citation: S.-F. Barrera-Molano, J.-E. Méndez-Gómez, D.-Z. Salek-Chaves, "Software Integration of an IMU Sensor to a CubeSat Platform Based on CSP (CubeSat Space Protocol)," *Revista Facultad de Ingeniería*, vol. 32, no. 64, e15732 2023. https://doi.org/10.19053/01211129.v32.n64.2023.15732

Abstract

The development and use of nanosatellites have increased in recent years. Space programs were exclusive to governments with significant capital. Now, nanosatellites

[1] Fuerza Aérea Colombiana (Cali-Valle del Cauca, Colombia). sergio.barrera@fac.mil.co. ORCID: 0000-0003-1140-7634

[2] Fuerza Aérea Colombiana (Cali-Valle del Cauca, Colombia). javier.mendez@fac.mil.co. ORCID: 0000-0002-2410-5400

[3] M. Sc. Fuerza Aérea Colombiana (Cali-Valle del Cauca, Colombia). dib.salek@fac.mil.co. ORCID: 0000-0002-1903-1910

have changed the orbital ecosystem and have allowed new regions and a wide range of industries to position themselves in the so-called new space, a growing sector that democratizes the commercialization of space thanks to a smaller, more agile, and affordable technology. Nanosatellites have similar capabilities to their conventional counterparts but are commonly used for highly specific missions such as Earth observation, telecommunications, and meteorology. The complex development of a nanosatellite requires solving different engineering problems to assembly and integrate all hardware and software components into a small space. It requires the monitoring, control, and operation of the satellite's features from the ground segment. One major challenge is the development of satellite mission control software for both space and ground segments in limited-time scheduled missions. This article describes the integration process of an EPSON M-G364PDCA Inertial Measurement Unit (IMU) to a CubeSat platform nanosatellite based on the Danish company GomSpace devices and a CSP protocol. The result is a hardware implementation and software development by the Colombian Air Force (FAC) team as part of the FACSAT-2 program. The integration of these components into the space and ground segment contributes to solve one challenge in the development of control software for space missions, as described above, and becomes the first approach for Colombian space nanosatellites software development. In addition, this research presents the Colombian Air Force configuration for space mission subsystems on the CSP network, the software development for the main on-board computer based on a NanoMind A3200 to setup the IMU —controlled and monitored from the ground segment through a CSP terminal on a Linux server—, and setup telemetry data from space to be sent periodically to the ground segment and stored locally in a MongoDB database for its subsequent visualization and analysis.

Keywords: CSP; integration; IMU; space mission; software; test; validation.

Integración de software de un sensor IMU a una plataforma CubeSat basada en una red CSP (Protocolo Espacial CubeSat)

Resumen

El desarrollo y uso de nanosatélites ha incrementado en los últimos años. Los programas espaciales eran territorio exclusivo de un puñado de gobiernos con un importante capital. Ahora los nanosatélites han cambiado el ecosistema orbital y han permitido que nuevas regiones y una amplia gama de industrias se posicionen en el llamado nuevo espacio, un sector en crecimiento que democratiza la comercialización del espacio gracias a una tecnología más pequeña, ágil y asequible. Los nanosatélites tienen capacidades similares a sus contrapartes convencionales, pero generalmente se usan para misiones muy específicas, como la observación de la Tierra, las telecomunicaciones y la meteorología. El complejo desarrollo de un nanosatélite demanda resolver diferentes problemas de ingeniería para ensamblar e integrar todos los componentes en un pequeño espacio a nivel de hardware y software, que permita el monitoreo, control y operación de las características del satélite desde el segmento tierra. Un reto importante es el desarrollo de software de control de misiones satelitales para los segmentos espacial y terrestre en misiones programadas con tiempo limitado. Este artículo describe el proceso de integración de la Unidad de Medición Inercial (IMU) EPSON M-G364PDCA en un nanosatélite de plataforma CubeSat basado en los dispositivos de la compañía danesa GomSpace y el protocolo CSP. El equipo de la Fuerza Aérea Colombiana presenta una implementación de hardware y desarrollo de software como parte del desarrollo del programa FACSAT-2. La integración de estos componentes en el segmento espacial y terrestre contribuye a resolver uno de los retos importantes en el desarrollo del software de control de misión para misiones espaciales descrito anteriormente, y se convierte en el primer enfoque para el desarrollo de software propio para nanosatélites espaciales. Además, la presente investigación en la Fuerza Aérea Colombiana garantiza una perspectiva diferente a partir de la configuración de los subsistemas que forman parte de la misión espacial en la red CSP, el desarrollo del software para la computadora principal a bordo —basado en un NanoMind A3200 que permite configurar, controlar y monitorear la

IMU desde el segmento terrestre a través de un terminal CSP en un servidor Linux— y configurar los datos de telemetría desde el espacio para enviarlos periódicamente al segmento terrestre y almacenarlos localmente en una base de datos MongoDB para su posterior visualización y análisis.
Palabras clave: CSP; IMU; integración; misión espacial; software; test; validación.

Integração de software de um sensor IMU a uma plataforma CubeSat baseada em uma rede CSP (CubeSat Space Protocol)

Resumo

O desenvolvimento e uso de nanossatélites tem aumentado nos últimos anos. Os programas espaciais eram território exclusivo de um punhado de governos com capital significativo. Agora os nanossatélites mudaram o ecossistema orbital e permitiram que novas regiões e uma ampla gama de indústrias se posicionassem no chamado novo espaço, um setor em crescimento que democratiza a comercialização do espaço graças a uma tecnologia menor, mais ágil e acessível. Os nanossatélites têm capacidades semelhantes às suas contrapartes convencionais, mas geralmente são usados para missões muito específicas, como observação da Terra, telecomunicações e meteorologia. O complexo desenvolvimento de um nanossatélite exige a resolução de diferentes problemas de engenharia para montar e integrar todos os componentes num pequeno espaço ao nível do hardware e do software, que permita a monitorização, controlo e operação das características do satélite a partir do segmento terrestre. Um desafio importante é o desenvolvimento de software de controle de missão de satélite para os segmentos espacial e terrestre em missões programadas com tempo limitado. Este artigo descreve o processo de integração da Unidade de Medição Inercial EPSON M-G364PDCA (IMU) em um nanossatélite da plataforma CubeSat baseado nos dispositivos da empresa dinamarquesa GomSpace e no protocolo CSP. A equipe da Força Aérea Colombiana apresenta uma implementação de hardware e desenvolvimento de software como parte do desenvolvimento do programa FACSAT-2. A integração desses componentes no segmento espacial e terrestre contribui para resolver um dos importantes desafios no desenvolvimento de

software de controle de missão para missões espaciais descritas acima e se torna a primeira abordagem para o desenvolvimento de software proprietário para nanossatélites espaciais. Além disso, a presente investigação na Força Aérea Colombiana garante uma perspectiva diferente desde a configuração dos subsistemas que fazem parte da missão espacial na rede CSP, o desenvolvimento do software para o computador de bordo principal —baseado em um NanoMind A3200 que permite configurar, controlar e monitorar a IMU do segmento terrestre através de um terminal CSP em um servidor Linux - e configurar os dados de telemetria do espaço para serem enviados periodicamente ao segmento terrestre e armazenados localmente em um banco de dados MongoDB para sua posterior visualização e análise.

Palavras-chave: CSP; IMU; integração; missão espacial; programas; teste; validação.

I. INTRODUCTION

Currently, space missions face the challenge of releasing solutions with high-quality standards in short periods of time [1]. As a result, most space companies seek to improve their processes by developing devices with high-end hardware running a Software Development Kit (SDK) that contains the main common features across different satellites. It allow them to use the same products and software to reduce the development time and limits the development of the control software to meet the specific requirements of the mission [2]. GomSpace S/A is a manufacturer of nanosatellites with customers in sectors such as defense, academic, government and commercial markets. They manufacture devices like On-Board Computer (OBC) understood as the brain of the satellite [3], radio links, Attitude Determination And Control System (ADCS), “based on implementing the controller and sensor subsystems to monitor attitude determination” [4], and software integrations into the customer’s space mission requirements [1].

GomSpace products include software and hardware based on a CubeSat Space Protocol (CSP) that enables distributed embedded systems to deploy a service-oriented network topology. It is designed for embedded systems such as the 8-bit and 32-bit ARM/AVR microcontrollers from Atmel, designed for space segments in On-Board Computers and different subsystems with low power consumption. The implementation is written in C programing language and is ported to run on FreeRTOS, POSIX, and threads-based operating systems such as Linux for ground segments [1].

The Colombian Air Force (FAC by its Spanish acronym) developed an on-job training program based on a transfer of knowledge with the company GomSpace A/S. Their software/hardware environment were used to integrate an Inertial Measurement Unit as an external hardware device operated by the internal On-Board Computer. Thus, it was possible to share IMU’s features as measurement of acceleration and velocities rates across the CSP network in space to ground as a service from the OBC. This development is designed as a part of the FACSAT-2 program future space missions [2].

The software development to integrate a third-party hardware on a nanosatellite using GomSpace utilities is based on the A3200 command & management SDK designed for the NanoMind A3200 / AVR32 platform [5]. The ground segment uses the MS100 Command & Management SDK [6] running on a Linux server. These 2 segments communicate with a RF link through UHF AX100 radios on both, as any satellite in space will perform, thus allowing us to use different functionalities as telemetry from space to ground, performing IMU control, “remote monitoring and control of spacecraft subsystems and payloads” [7] as a regular satellite will operate.

II. METHODOLOGY

The development is given by a deductive methodology, organized in a step-by-step process, as described in Fig. 1.

Fig. 1. Diagram of the methodology.

A. Hardware Analysis

The development is based in two segments: space and ground. Each segment plays a crucial role in the overall functioning of the satellite system, and they must all work together seamlessly to ensure a successful operation of the satellites and the services they provide.

Space segment is composed by seven subsystems designed as normal nanosatellite with standard features, including one NanoCAM C1U as payload and one NanoMind A3200 as the main On-Board Computer, the subsystems are:

- A3200 - OBC
- A3200 - ADCS
- P60 - EPS (2 extra nodes)
- ACU - Solar panel control unit
- PDU - Power distribution control unit
- AX100 - UHF radio link Communication

- NanoCAM – Payload

The NanoMind A3200 is the main computer on board. It is composed of a high performance AVR microcontroller equipped with flash devices and sensors. Fig. 2 shows a top view of the OBC A3200 and highlights its compact size and its two lateral connection ports for the AVR programmer, which can be used for GOSH or for KISS serial connection [5]. Its main function is to control and coordinate all subsystems on board the satellite, including the payload, communication and power systems [5]. In this research, the OBC controls and operates the IMU sensor to include their features to the satellite system.

Fig. 2. NanoMind A3200 - On-Board Computer, top view [11].

1) IMU M-G364PDCA Hardware. The M-G364PDCA is a small form factor Inertial Measurement Unit (IMU) with 6 degrees of freedom, angular velocities, and triaxial linear accelerations. Additionally, it provides high stability and high precision measurement capabilities using high-accuracy offset technology. A variety of calibration parameters are stored in the IMU's memory and are automatically reflected in the measurement data sent to the application after the IMU is powered on. Fig. 3 shows the functional block diagram of the IMU M-G364PDCA [8].

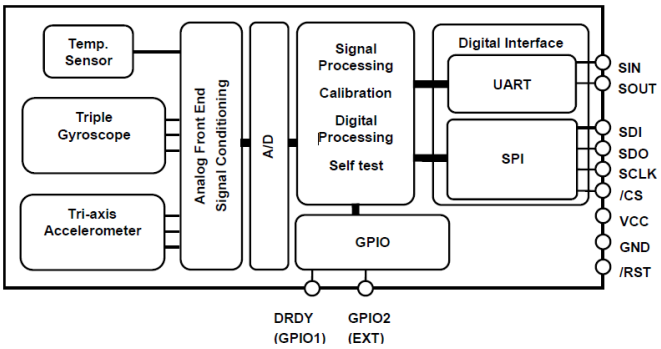


Fig. 3. IMU functional block diagram. Source: EPSON.

2) Ground Segment. "A ground system supports the space segment … and relays to users mission data generated by onboard instruments and received from the space" [9]. The ground segment is composed of 01 UHF radio link AX100 subsystem controlled by a Linux server running the mission control software [6]. This tool is critical for managing and controlling satellites, some of the key functionalities of this software include:

- Telemetry Data Processing: The software can collect, process, and analyze telemetry data transmitted by the satellite, thus providing important information about its health and status, and store it in a MongoDB for future access.
- Commanding and Control: Mission control software can send commands to the satellite and adjust its functionalities or operating parameters as needed.
- Payload Management: Satellite mission control software can manage the operation and performance of the satellite's payload, such as cameras, sensors, and scientific instruments.
- Fault Detection and Diagnosis: The software can detect and diagnose faults in the satellite's systems and provide recommendations for corrective actions.
- Communication Management: The software must manage the communication links between the satellite and the ground station, ensuring that data is transmitted and received correctly.

B. Software Analysis

Satellite systems require various types of software to perform different functions and support different subsystems, oriented by the ECSS standards, “ESA Council adopted a resolution that confirmed the Agency’s commitment to transferring the existing system of ESA space standards to a new set of standards that were to be prepared by the European Cooperation for Space Standardization (ECSS)” [10] [11]. This research aims to develop two key software components required for the satellite system:

- Flight software: This software controls the behavior of the satellite in orbit and executes the various subsystem commands, including the IMU control and operation. “The FSW is responsible for the calculation of satellite position/orbit, operation of scientific instrument payload, and communication with ground antenna” [12]
- Ground software: This software manages the communication between the satellite and the ground station, including data transmission, reception, and processing.

The flight software running in the OBC in this research focuses on controlling and operating the IMU features on the satellite system. Regarding software requirements, the IMU demands using SPI serial interface to access the control, data, and measurement parameters information; the read/write operations must be programed on the OBC for the satellite system to have full access.

The IMU starts immediately after a hardware restart or power-up, the hardware begins internal initialization; at that moment, all the values of the registers and states of the external pins are not defined. Once it is completed, the device goes into configuration mode; then, it is possible to adjust the operating values of the IMU. When the configuration is completed, it switches to sampling mode to read the temperature, angular velocity, and acceleration data in all three axes.

The ground software runs on a Linux desktop with Ubuntu distro. It needs to communicate with the satellite by accessing the user to request information from the satellite, sending commands, receiving, and managing the telemetry the satellite

sends periodically, and storing the information in a MongoDB data base [13], both flight and ground software are oriented by the ECSS-E-ST-40C [14].

C. CSP Services

This section presents information related to the CubeSat Space Protocol (CSP) network services and the implementation in the NanoMind and the Linux Server to include the IMU features on the system. The protocol is intended to provide a standard way for CubeSats to communicate with each other and with ground stations on Earth [1].

The CSP network defines a set of packet formats, data structures, and transmission procedures optimized for the small size and limited power and computing resources of CubeSats. The protocol supports both point-to-point and broadcast communications and can be used for a range of data types, including telemetry, images, and command and control messages. The application in this research is limited to [15]:

- Allow communication between the mission subsystems in space and ground segments, routing tables in every subsystem for intercommunication, and hops between segments using the RF AX100 radios as a gateway in both satellite segments.
- Implementations for space and ground segments software with the A3200-SDK and MS100-SDK libraries and structure. This is basic to include the software development for implementation of new service for the IMU features into the CSP network.
- Services information is required to request data from the IMU, store in a parameter table accessible through the CSP network, parameters are sampled (updated) periodically and others are updated if the user samples the sensor manually with a CSP command.
- Services related to telemetry information are sent periodically from the space to ground segment and then included in a local database.

The implementation of a new CSP service protocol requires the programming of the server and the client, respectively. Additionally, a relationship must be maintained

between the sent packets and those that are expected to be received through CSP to ensure correct communication [16]. The implementation of a CSP command has the following characteristics:

CSP-Term Client Commands Register: Commands and subcommands must be registered within the CSP-Term console by the user, so that when the user requests either the help, autocomplete or the use of the command option, the help text appears on the screen of the command in a friendly way. Highlighting the operation of the help command would display the corresponding options and messages.

Once the invoked command completes its execution, the software handler is in charge of packaging the return data in the format defined by the command to ensure that both the client and the server handle the same data sizes and format. For example, reading the IMU will invoke the SPI function to obtain the sample, later, the handler must pack this data in the required format to send it back by CSP to the client that requested it; likewise, it will obtain the information in the same format for its interpretation.

D. Development

The development is based on the hardware, software, and CSP network research to include the IMU service in the software image for space segment in the On-Board Computer NanoMind A3200 [13]. Also, it aims at including the new IMU service and the configuration of the telemetry beacon from space to ground into the CSP terminal in the Linux server, also solving the next issues:

- CSP network configuration for each subsystem.
- Include the IMU service in the NanoMind A3200 image for space.
- Include the IMU parameter tables in the space segment on the OBC.
- Include the IMU service in the CSP-Terminal in the Linux server.
- Setup the telemetry beacons from space to ground IMU data.

The network segmentation is carried out following the recommendations of the CSP protocol, leaving the addresses of the space segment in the range of (0-23), and those of the ground segment in the range of (24-31). The CAN interface is taken as the main bus in the space segment (since all its nodes support this interface). The

routing of the network packets is designed to connect all nodes to each other, and it seeks to restrict the routing of packets outside this network segment through the AX100, and vice-versa, as a connection gateway between ground and space segments.

Moreover, the ground station runs a ZMQ interface defined as a communication interface between tasks to be executed on the Linux Server (CSP-Term, Beacon parser). In the same way, the USART interface (via KISS protocol) is selected as the main interface for the connection between the Linux server PC and the AX100 communication module, since it supports network hopping at level 0, thus allowing to route the packets from the ZMQ proxy to the satellite through the AX100 node. Fig. 4 shows the assigned addressing for each node on the proposed CSP network for space and ground segments.

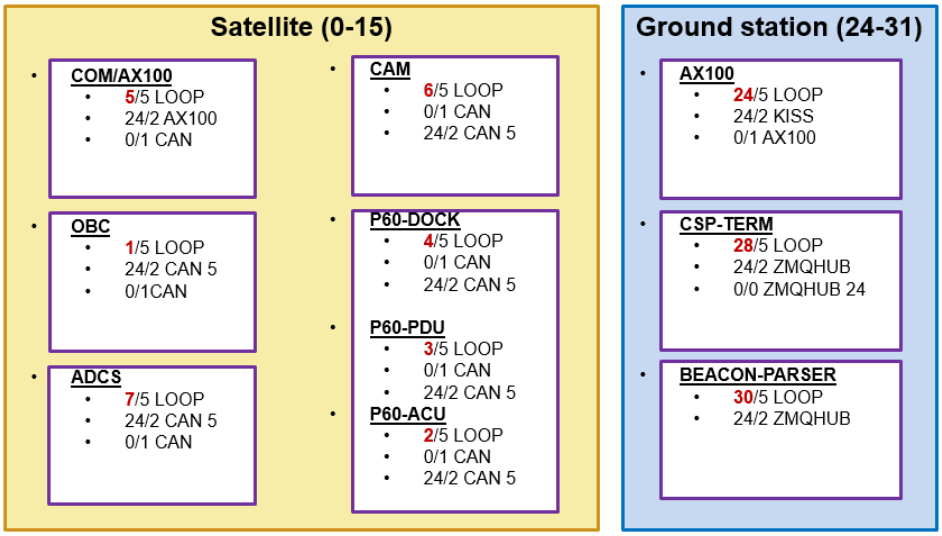


Fig. 4. CSP network and its routing within each subsystem.

For the adequate communication and functioning of the system, the FAC team had to develop a telemetry beacon parser. It's an application part of the CSP network with the purpose of receiving the packets from space to ground, which correspond to the CSP telemetry - housekeeping packets.

The telemetry beacon parser oversees identifying the beacon received, decompression and entry of the data (parameters) in the MongoDB database, as

can be seen in **¡Error! No se encuentra el origen de la referencia.**. In this way, the proposed configuration for the ground segment is fulfilled as described in Fig. 5, which shows the main log from beacon parser, receiving some beacons, decoding, parsed and added into the MongoDB with timestamp, type, version, sat_id to complete the information.

Table 1. Telemetry data stored in the MongoDB database.

Item	Parameter
1	init_status
2	auto_status
3	x_gyro
4	y_gyro
5	z_gyro
6	x_accl
7	y_accl
8	z_accl
9	temp

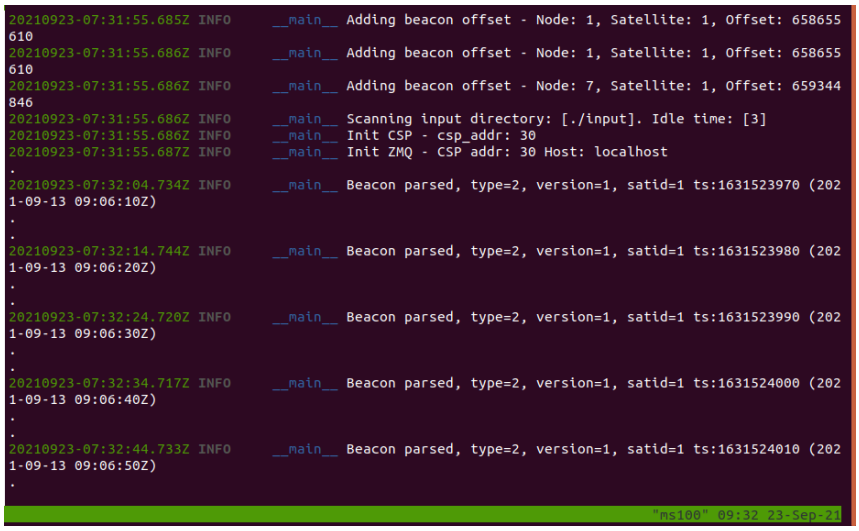


Fig. 5. Beacon parser receiving a CSP packet.

Likewise, some scripts are developed to ensure that all the processes requested on the server are setup prior to the correct execution and configuration. The final mission control software running in the Linux server is shown in Fig. 0, and presents the main four running applications: ZMQ Bridge, ZMQ Proxy, Beacon parser, and a

CSP terminal. The four are executed in parallel to send commands from the server by the user. The application tmux is used as a terminal multiplexer; it lets the user switch easily between several programs in one terminal, detach them (they keep running in the background), and reattach them to a different terminal without the risk of losing information during the switch, keeping the terminal in the background.

E. Integration and Testing

The integration phase development, the component assembly, and interconnection run in a cooperative way, to check that there are no adverse emergent behaviors. At the same time, they check the response of a complete system and how each component and subsystem "affect" the other ones. As described by the NASA "Product integration is the engineering of the subsystem interactions and their interactions with the system environments (both natural and induced). Also in this process, lower-level products are assembled into higher-level products and checked to make sure that the integrated product functions properly and that there are no adverse emergent behaviors" [17].

The testing stage can be divided into two: the first one defines the procedures to check that the new functionalities work properly; the second tests all functionalities developed to integrate the IMU into software and hardware in the nanosatellite from the CSP Terminal, verifying that the result complies with the requirements.

The Integration of the elements and their performance were validated with the technological partner: GomSpace. The next tests to perform on the system were defined, and the results after their execution were the following:

- Check power up scenario, setup, and communication between space and ground segment: PASS.
- Check power up scenario for the IMU, setup, set single sampling and auto sampling to retrieve data from the IMU to the CSP-Terminal, then power off the IMU and check power consumption: PASS.
- Check telemetry beacon from space to ground segment and the consequent analysis and store in the MongoDB, allowing to read the IMU data log in the Linux Server: PASS.

The execution of the test obtained an overall pass result, thus concluding that the resultant system meets the requirements. During the test execution, the next terminal output on the Linux server side looks as shown in Fig. 6.

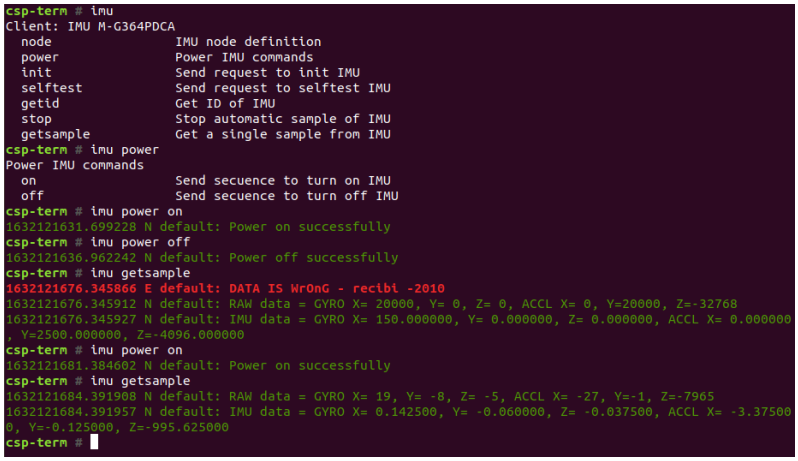


Fig. 6. CSP Terminal during the test execution.

The Figure 6 shows an error produced by trying to retrieve data from the IMU when the state is powered off, so the actual error is a correct result to the user. Later, the user powered on the IMU and tried to get a sample, resulting in correct data information retrieved from the satellite.

III. RESULTS

Using software analysis to include the A3200 SDK into the space segment and MS100 SDK, the IMU services and new features, into the ground segment, it is possible to establish and execute the IMU commands from the CSP-terminal. It allows the user to initialize the device, obtaining the serial, starting the auto-sampling task, or requesting an individual sample of the sensor successfully. An example of the IMU service is shown in Fig. 7 with all the commands available in the terminal.

```
csp-term # imu
Client: IMU M-G364PDCA
  node                IMU node definition
  power               Power IMU commands
  init                Send request to init IMU
  selftest            Send request to selftest IMU
  getid               Get ID of IMU
  stop                Stop automatic sample of IMU
  getsample           Get a single sample from IMU
csp-term # imu power
Power IMU commands
  on                  Send secuence to turn on IMU
  off                 Send secuence to turn off IMU
csp-term # imu power on
1632121631.699228 N default: Power on successfully
csp-term # imu power off
1632121636.962242 N default: Power off successfully
csp-term #
```

Fig. 7. CSP terminal output IMU service.

The complete setup for the space segment is shown in Fig. 8, and the communication RF link for the ground segment is shown in Fig. 9.

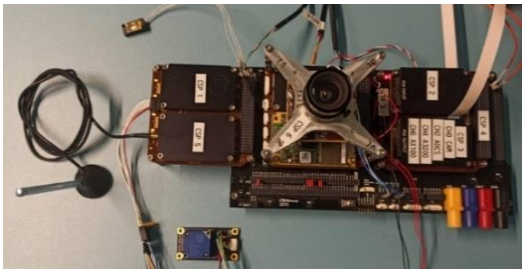


Fig. 8. Hardware Implementation of Space segment with IMU on nanosatellite.

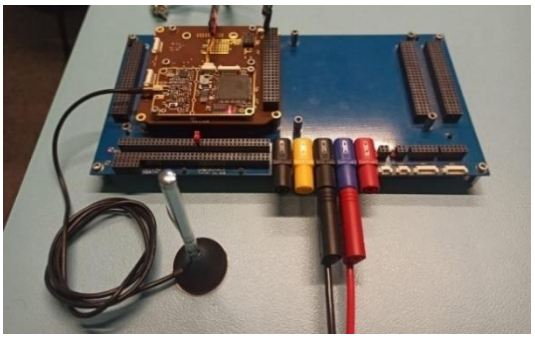


Fig. 9. Hardware Implementation of ground segment with Radio AX100.

Using the satellite with the software developed allows the user to control the IMU from the ground segment, accessing all the measurement features, updating the values in a parameter table that other devices can access from CSP services, and

a beacon sending information from the space and receiving the data on the ground segment and storing in a MongoDB data base. The complete terminal application is shown in Fig. 10 with a split screen running the four applications at the same time, i.e, CSP-Terminal, ZMQ Bridge, ZMQ Client, and Beacon parser.

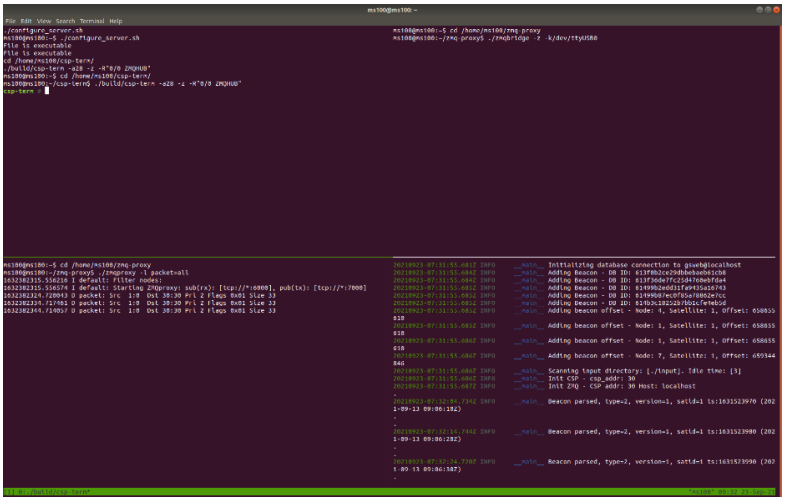


Fig. 10. Mission control software running CSP, ZMQ Bridge, ZMQ Proxy, and Beacon parser.

IV. Discussion and Conclusions

Once the EPSON IMU M-G364PDCA sensor integration has been completed at the software level with the OBC NanoMind A3200 and the CSP-Term command terminal using the different tools provided by GomSpace, it is possible to conclude:

- This is the first registered integration of an IMU with a space platform developed by a Colombian team, so the main achievement is the acquired capability to carry out these software engineering activities and the transfer of knowledge for space developments.
- The Colombian Air Force personnel involved in the development of the integration faced the challenge of understanding the CSP protocol. It enabled to integrate the IMU with other hardware components correctly into the satellite, and the creation of new CSP commands to control the IMU according to defined functionalities by the manufacturer. The CSP protocol is mandatory to any development under GomSpace product lines.

- The implementation of CSP commands requires a specific hardware with a unique structure of the protocol and packets to be sent and received by the subsystem at both ends of the communication, which requires prior planning to define these requirements and guarantee its correct operation and communication once implementation is complete.
- The satisfactory results of this process open the possibility of developing new applications on other types of integration and new projects for space platforms in the Colombian space industry lead by the FAC personnel in the near future, as well as new satellites or space systems.
- Specifically, for the FACSAT-2 mission, the knowledge acquired by developing these activities allows us to approach the integration and develop test procedures for the different subsystems that are part of the satellite, thus making the mission successful.

AUTHORS' CONTRIBUTION

Sergio-Fernando Barrera-Molano: Software, formal analysis, resources, investigation, visualization, validation, writing-original draft, writing-review & editing.
Javier-Enrique Méndez-Gómez: Software, formal analysis, resources, investigation, visualization, validation.
Dib-Ziyari Salek-Chaves: Investigation, formal analysis, methodology, writing-original draft, writing-review & editing.

ACKNOWLEDGMENTS

The authors thank the Colombian Air Force for the tools and support to develop the project, the Colombian government for the trust in the Colombian Space Program and research in this field, and finally to GomSpace for being the technological ally in the development of the satellite program of the Colombian Air Force.

FUNDING

This article is part of the results of the satellite mission FACSAT-2 "CHIRIBIQUETE" research project, part of the Colombian Air Force's space program.

REFERENCES

[1] GomSpace, *CubeSat Space Protocol (CSP)*, 2011. https://bytebucket.org/bbruner0/albertasat-on-board-computer/wiki/1.%20Resources/1.1.%20DataSheets/CSP/GS-CSP-1.1.pdf?rev=316ebd49bed49fdbb1d74efdeab74430e7cc726a#:~:tex

[2] Fuerza Aérea Colombiana, *ENGg–SW-STS L0-001-V 1.0*, Santiago de Cali, 2021.

[3] H. E. H. Ahmed, E. Kamal, A. Elsayed, "Telemetry microcomputer application in satellites OBC," in *First Asian Himalayas International Conference on Internet, Kathmundu*, Nepal, 2009, pp. 1-6. https://doi.org/10.1109/AHICI.2009.5340347

[4] S. Mohamed, E. Ahmed, "Design and Implementation of an ADCS for a CubeSat. Journal of Engineering Research," *Journal of Engineering Research,* vol. 7, no. 2, pp. 15-21, 2023.

[5] GOMSPACE, *NanoMind A3200,* Aalborg East: GomSpace, 2021.

[6] GOMSPACE, *NanoCom MS100,* Aalborg East: GomSpace, 2021.

[7] ECSS, *ECSS-E-ST-70-41C - Telemetry and telecommand packet utilization*, Netherlands, 2016.

[8] Seiko Epson Corporation, *M-G364PD High Stability IMU (Inertial Measurement Unit),* Tokyo, 2022.

[9] W. J. Larson, J. R. Wertz, *Space Mission Analysis and Design*, El Segundo, CA: Microcosm Press - Kluwer Academic Publishers, 1999.

[10] M. Jones E. Gomez, A. Mantineo, U.K. Mortensen, "Introducing ECSS Software-Engineering Standards within ESA," *ESA Bulletin ,* no. 111, p. 8, 2002.

[11] ECSS, *European Cooperation for Space Standardization*, 2023. https://ecss.nl/

[12] R. Sandau, R. P. Hans, A. Valenzuela, *Small Satellites for Earth Observation*, Springer, 2008.

[13] S. Apel, C. Kastner, "An Overview of Feature-Oriented Software Development," *The Journal of Object Technology,* vol. 4, e1, 2009.

[14] ECSS Secretariat, *ECSS-E-ST-40C –Space engineering / Software*, Netherlands, 2009.

[15] E. J. Riis, *Design and Implementation of a Reliable Transport Layer Protocol for NUTS*, Trondheim: Norwegian University of Science and Technology, 2015.

[16] M. Zabala, L. Cuenca, J. León, F. Cabrera, "Arquitectura de acoplamiento entre INS/GPS para navegación precisa en trayectorias establecidas," *Maskay,* vol. 8, no. 1, pp. 13-19, 2018.

[17] NASA, *NASA Systems engineering handbook*, Washington D.C., 2007.