Visual based GNC system from prototype to flight software

Florin-Adrian STANCU*,[1], Víctor Manuel MORENO VILLA[2],
Carlos DOMÍNGUEZ SÁNCHEZ[2], Andrei Valentin PLAMADEALA[1],
Daniel OVEJERO PROVENCIO[2]

*Corresponding author
[1]GMV Innovating Solution SRL, Skytower,
246C, 32nd floor, district 1, 014476, Bucharest, Romania,
fstancu@gmv.com*, andrei.plamadeala@gmv.com
[2]GMV Aerospace and Defence SAU,
Isaac Newton, 11 P.T.M, Tres Cantos, 28760, Madrid, Spain,
vimoreno@gmv.com, cdsanchez@gmv.com, daniel.ovejero@gmv.com

DOI: 10.13111/2066-8201.2023.15.1.9

Received: 05 October 2022/ Accepted: 19 January 2023/ Published: March 2023

Abstract: *Juventas is a 6U CubeSat which is part of HERA planetary defense mission and takes advantage of an innovative visual based guidance, navigation and control (GNC) system to perform autonomous navigation in the proximity of the Didymain system. The GNC system is designed to ensure safe navigation in the harsh and unpredictable environment of deep space and finally to take more risks by landing on the surface of Dimorphos. To achieve a qualitative software (SW) product, a dedicated procedure of SW lifecycle is developed by starting with GNC and image processing design, which concludes with the final embedded system that will perform the visual navigation task. A Design, Development, Verification and Validation (DDVV) approach is developed to achieve the flight software: model in the loop (MIL), auto-coding, software in the loop (SIL), processor in the loop (PIL) and finally hardware in the loop (HIL). The DDVV is developed by having as guideline the ECSS standards for SW. The flight software reaches maturity by performing dynamic/static code analysis and code coverage. To ensure an optimal process, a waterfall life-cycle is considered, where dedicated MIL, SIL, PIL and HIL test benches are developed to fully support the activity and to reduce to minimum the development costs.[7]*

Key Words: *guidance navigation and control, visual navigation, model in the loop, auto-coding, software in the loop, processor in the loop, flight software*

1. INTRODUCTION

HERA is a planetary defense mission developed by the European Space Agency with the objective of characterizing the Didymos system, after the impact of the DART spacecraft with Dimorphos.

The objective is to demonstrate and validated asteroid deflection technology, part of the Asteroid Impact and Deflection Assessment (AIDA) international collaboration.

Didymos binary asteroid system is composed of two bodies: Didymos and Dimorphos (orbiting around Didymos). HERA will carry two CubeSats, scheduled to be released in the vicinity of the Didymos system and perform missions with a higher degree of risk.

One of the CubeSat is Juventas, designed around CubeSat standards with the dimension of 6U. GMV is in charge to design and integrate the software (SW) implementation of the visual based GNC system of the Juventas satellite.

A functional Engineering Simulator (FES) is developed in Matlab/Simulink, [5], where the GNC model and image processing (IP) take shape.

Matlab proved to be an optimal solution to integrate complex DDVV approaches with low costs.

The DDVV plan is the main strategy to build a qualitative and reliable code to finally reach the status of flight code using extensive SW verification and Validation (V&V). The DDVV is based on auto-coding techniques for GNC and manual coding for IP.

GNC is a complex system that can be developed in parallel by multiple engineers, thus being subjected to considerable changes during development it proved to be a good candidate for auto-coding techniques.

The IP algorithms show computational complexity and can require a considerable amount of resources to be run on a dedicated space processor.

Therefore, the IP is considered to be manually coded to take full advantage of SW optimization possibilities.

2. JUVENTAS VISUAL BASED GNC

Juventas is a 6U CubeSat with the mission objective to gather scientific information by performing the characterization of the Didymos system. To perform accurate navigation autonomously, a complex visual based GNC system is essential to cope with the challenging operational environment of the Juventas mission.

The GNC system is responsible to ensure safe navigation by performing sensor measurement fusion.

One of the key elements is the image processing technology that is in charge to process onboard images to extract navigation information, used by GNC to drastically increase navigation performances.

The GNC and IP elements are component elements of GNC Application SW (ASW), which are integrated into a Linux process called GNC ASW process (as described in chapter 4). The internal architecture, inputs and outputs (I/O) of GNC ASW are depicted in Fig. 1 and widely described in [1], being designed to map the principal functionalities:

1. GNC mode manager module where GNC modes are selected,
2. Measurements processing module where sensor inputs are pre-processed,
3. Navigation module where the main element is the navigation filter,
4. Command module where commands are computed,
5. FDIR module where GNC fault, detection, isolation and recovery elements are implemented.

The IP technology is inherited from HERA, [2], which are image processing algorithms designed for space asteroid space missions to detect the line of sight (LOS) towards asteroids:

1. Center of Brightness (CoB) with Didymos masking where the position of Dimorphos in the image frame is estimated.
2. Maximum correlation with Lambertian sphere (LAMB) where the center of Didymos asteroid is estimated in the image frame.

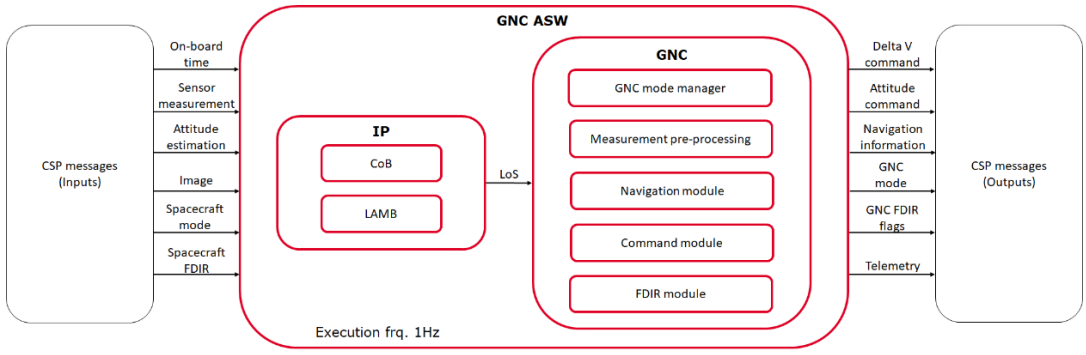

Fig. 1 – GNC ASW architecture

3. SOFTWARE LIFECYCLE DEFINITION

In this chapter is described an innovative SW lifecycle used to develop the GNC ASW of Juventas CubeSat from design until flight code.

The approach is following ECSS-E-ST-40C and ECSS-Q-ST-80C standards and additionally ECSS-E-HB-40A handbook, with the comment that are tailored with the objective of obtaining a qualitative code for CubeSats, when the mission objective fulfillment does not depend on the CubeSat.

The SW lifecycle is developed around the phases defined in ECSS-E-ST-40C: SW specification, SW design, SW implementation and SW V&V.

The DDVV strategy for Juventas CubeSat is develop to fulfill SW criticality C. However, the DDVV strategy can be easily adapted to fulfill SW criticality B, the differences being presented in chapter 3.

The DDVV is a complex process composed of multiple elements, modeled as a waterfall lifecycle. Fig. 2 depicts the detailed elements of the DDVV strategy, which is based on the following main steps:

- Mission analysis and GNC/IP SW specification derivation: the mission constraints are analyzed and the mission and SW requirements are derived, starting from the main requirements of Juventas CubeSat defined at the system level.
- Development of FES: based on requirements the GNC and IP algorithms are designed together with environmental and sensor models. At this stage, the GNC is designed using a Simulink model-based approach by following modeling rules to ensure compatibility with the embedded coder toolbox of Matlab/Simulink, [6]. The IP is manually codded directly in C.
- Autocoding of GNC: the GNC model based is autocoded using an embedded coder toolbox to generate C code, [6]. Also, the GNC generated C code is integrated into the FES model for verifications activities.
- Integration on the emulated processor: the GNC and IP are integrated under a single Linux process called the GNC SW process. Furthermore, code validation and verification activities are carried out.
- Integration on real processor target: the GNC SW process is integrated into the real target. Furthermore, more code V&V activities are carried out.

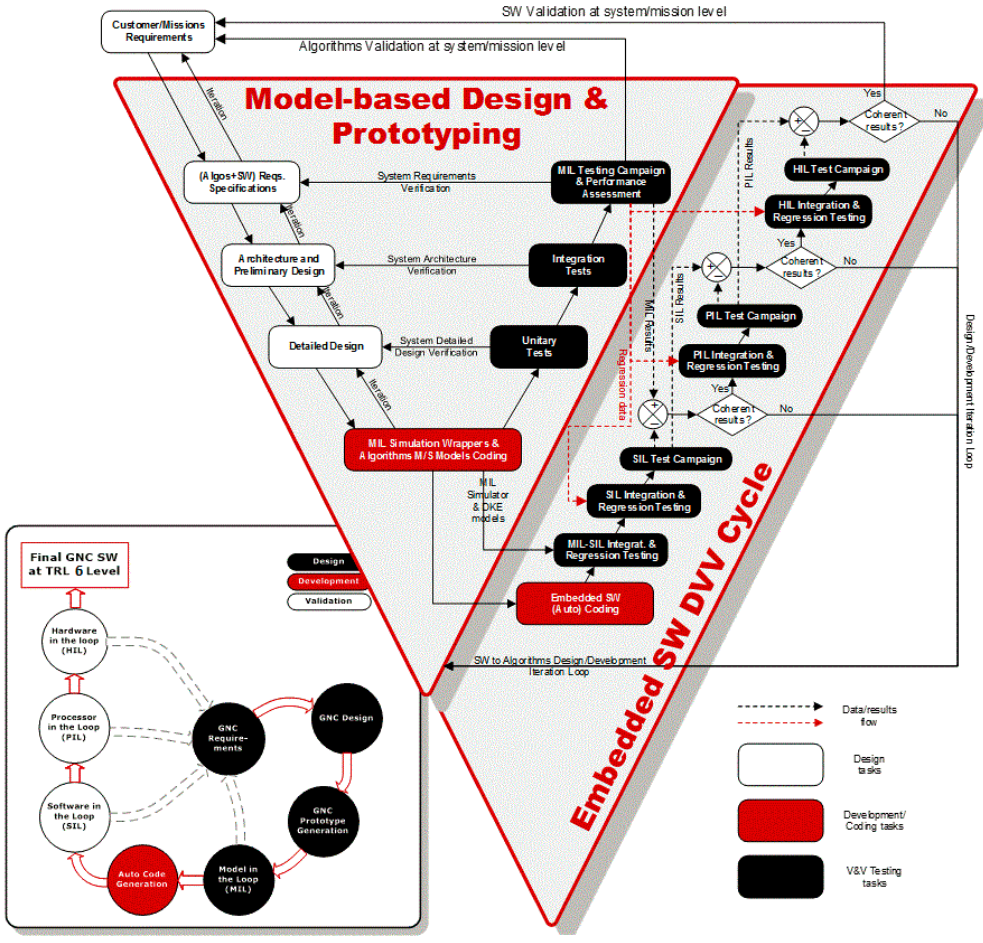

Fig. 2 – DDVV architecture, [3]

The DDVV strategy is supported by dedicated test benches, following the GNC ASW life cycle:

MIL → Auto-coding → SIL → EM-PIL → HIL

As can be observed the GNC ASW lifecycle integrates all the elements of the DDVV by means of test benches, designed in such a way to support the GNC and IP from prototyping until final implementation on the HW target. Furthermore, the test benches support the V&V for the entire GNC ASW lifecycle.

The V&V is a key process that allows to validate the GNC and IP performances and verifies the requirements at different stages of the GNC process lifecycle. The V&V is present at every GNC process lifecycle in order to detect possible issues in the early stages of development and correct them with minimal development costs. Furthermore, this approach allows us to gain full confidence in the implementation when moving to another stage (e.g. performing EM-PIL tests with data generated from SIL). At this point, the interconnection between all GNC process life cycle elements can be observed, being depicted as well in Fig. 2. When SW issues (or possible improvements) are reported, should be corrected at the MIL level and the process lifecycle is repeated. MIL test bench is used to perform unitary and integration tests to validate the algorithms. The SIL integration test of the C code provides

design feedback in the early stages of development. As a consequence, the best stage when issues can be detected to reduce development costs is during MIL and then at the SIL level.

The MIL objective is to design the GNC using the Simulink model component base approach and integrate the IP. This is the step when the GNC together with the IP are designed in accordance with requirements and functional performances are evaluated by means of extensive Monte Carlo (MC) simulations. Also, in the frame of MIL, the GNC/IP SW architecture and I/O are defined, which is formally defined as GNC ASW, Fig. 1. Furthermore, the mission environment and sensors are modeled in the Real World block to increase simulation fidelity. Moreover, to evaluate the feasibility of the mission and mission requirements fulfillment, an extensive mission analysis phase is carried out before starting the MIL implementation. As can be noticed the MIL is one of the most complex phases which provides as output the FES, considered to be the basis of the entire SW lifecycle.

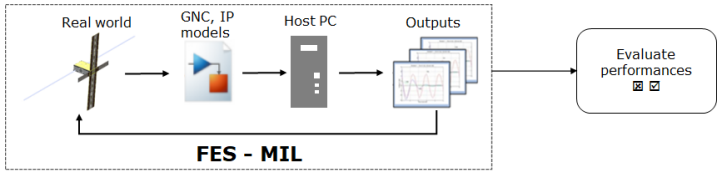

Fig. 3 – Model in the loop

The SIL step marks the integration of GNC auto-coding, generated from the MIL model by using Embedded Coder to generate ANSI C code. The auto-coding approach gives the possibility to correct design problems in the early stages of GNC design, thus reducing development costs. The IP is manually codded to fully optimize the code and to reduce at minimum the execution time and the memory allocation. The SIL takes advantage of the MIL architecture, by replacing the GNC and IP with S-functions to wrap the C code and still using the Simulink environment to perform MC campaigns. To increase the representativeness of SIL and bring the test bench as close as possible to final integration, the GCC compiler is used to compile the GNC and IP.

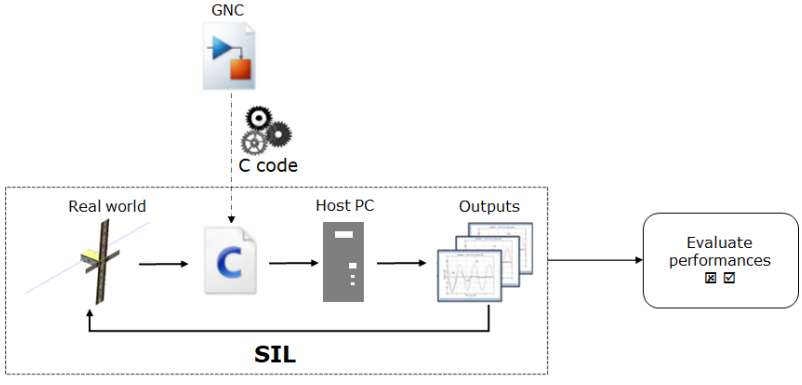

Fig. 4 – Software in the loop

The Emulated-PIL (EM-PIL) step marks the integration of the GNC and IP code into an emulated environment of the final target processor. The EM-PIL is a dedicated environment set up to be representative concerning the real processor and using simulated environmental conditions. The environmental conditions are generated using SIL by pre-recoding the input and output data. The inputs are read by EM-PIL and sent via CSP messages to the GNC ASW process to be run in Linux. The outputs of the GNC ASW process are compared with the

results of the SIL environment. It has to be mentioned that due to numerical precision, small differences can arise. Preliminary evaluation of SW performances is evaluated in order to see if the SW process fulfills the SW constraints, defined by requirements. The EM-PIL will provide valuable information regarding scheduling, worst-time execution and CPU load.

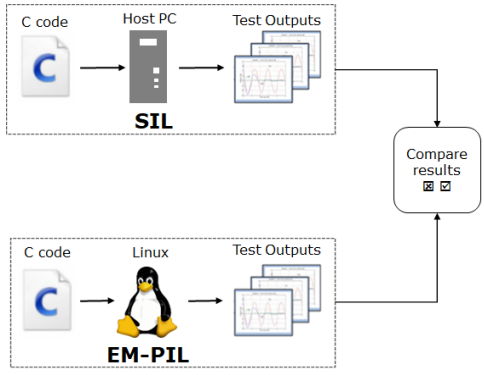

Fig. 5 – Emulated processor in the loop

The Hardware-in-the-Loop (HIL) marks the integration of the GNC ASW process in the real HW, represented by NanoMind Z7000. The same reference input and outputs used in EM-PIL are used. This marks the integration in the Juventas onboard computer by linking the GNC SW process with other processes. A dedicated visual based test bench is built, where the navigation camera captures representative images (generated using the SIL test bench) displayed on a high-definition screen and sent to the NanoMind Z7000. It has to be mentioned that a system integration test with the entire Juventas configuration will be performed at the system level by the Juventas responsible.

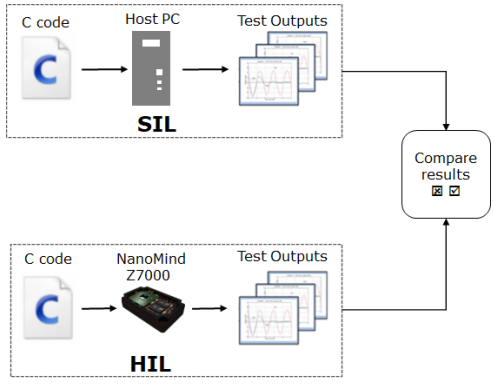

Fig. 6 – Hardware in the loop

4. SW DEVELOPMENT APPROACH

The GNC ASW is obtained using auto-coding and manually coding techniques. The Juventas GNC ASW follows a standard approach by tailoring ECSS-E-ST-40C and ECSS-Q-ST-80C. In consequence, rules are established to generate qualitative flight code by using auto-coding and manual coding.

The GNC is a model-based development where rules and guidelines are considered to ensure compatibility with the embedded coder to generate C code. The rules are based on the

SAVOIR auto code working group, [4], and can be divided into two main categories, [2]: modeling architectural and design rules and modeling implementation.

The modeling architectural and design rules are established to properly define the proper SW architecture of the GNC model.

The main aspects that have to be considered are:

- Clear architectural division between GNC models and other functionalities of FES.
- Clear definition of GNC interfaces.
- Avoid Simulink algebraic loops. Ensure data integrity when using rate transition block between modules.
- Define clear internal architecture of GNC. The internal architecture in the case of Juventas GNC is divided considering the functionalities of the modules, see Fig. 1.
- Set GNC model parameter tunability. The tunable parameters are foreseen to be updated during run-time.
- Embedded Coder setting for code production.

The modeling implementation rules define rules or guidelines to be followed at Simulink implementation of the GNC model.

The main aspects that have to be considered are:

- Modeling coding rules to prevent errors, non-optimized code and forbidden constructs. The embedded coder is selected as the system target file, ert.tlc. The internal architecture has to be defined by setting modules as atomic subsystems and avoiding infinite loops. Furthermore, functionalities that are used multiple times in the code have to be identified to generate common Simulink libraries.
- Model style rules to use a uniform coding style, complexity restriction and ensure readability. A uniform nomenclature of I/O or tunable/ non-tunable parameters is established. The model complexity has to be reduced to ensure SW complexity metrics, Table. 1. From a readability point of view, code comments inside functions can contribute significantly.

The auto-coding phase can start as soon as the modeling architectural and design rules are integrated into the FES and preliminary V&V is performed at the MIL level. During the code generation, the SIL block is automatically generated and ready to be embedded in the Simulink model, Fig. 7.

The IP is manually handwritten code directly in C by following similar coding and architectural rules as in the GNC case:

- Clear definition of IP interfaces.
- Define clear internal architecture of IP. The internal architecture in the case of IP is divided considering code functionalities.
- Set IP tunable parameters. The tunable parameters are foreseen to be updated during run-time.
- Code functionalities that are used multiple times in the code have to be identified in order to generate common functions.
- Uniform nomenclature of I/O or tunable/non-tunable parameters is established. The model complexity has to be reduced to ensure acceptable SW complexity metrics, Table. 1. From a readability point of view, code comments inside functions can contribute significantly.

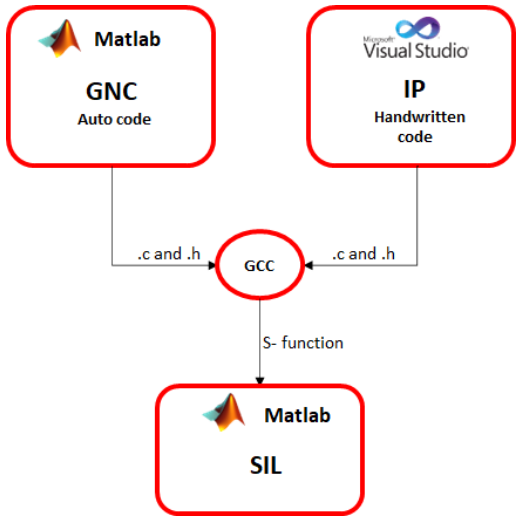

Fig. 7 – GNC ASW generation

The MISRA C: 2012 standard is proposed, via tailoring, as a guideline for GNC ASW, to guarantee the quality of the code and add consistency and safe use of the C code. The tailoring of the standards has to be performed taking into account the project needs.

In Table 1 is presented a summary of the SW elements used to build the GNC ASW as flight code.

This list is considered to be the minimum necessary to build a category C CubeSat flight SW and can be modified as needed. To reach category B, the list has to be updated by: the code coverage has to be 100% and independent SW Verification and Validation shall be considered.

Table 1 – SW checklist

SW Check Element	*Comments*
SW code category definition	Decided via reliability, availability, maintenance and safety analysis
Requirements definition	Defined at the system level and consolidated at the GNC ASW level
SW architecture definition	Top and lower level SW architecture definition
I/O list	Well establish I/O list, defined via interface control document: name, data type, dimension
Tunable parameter list	Well establish tunable parameter list, defined via interface control document: data type, dimension
HW target definition	Defined at the system level
Compiler definition	Defined at the system level
SW technical budget	Defined at the system level
Unitary testing	Performed at a lower level of SW architecture
Integration testing	Integrate lower-level function
System testing	Performed at the system level and evaluate latency of the outputs from the software are always within the specification no matter what the input conditions are applied

Fault injection	Determine how badly the software can behave if it is faulty and see how effective its fault-tolerance mechanisms are
Protected mathematical library	SW protections against division with zero, trigonometric functions, etc.
Code inspection	For manual codding
Static code check	Check for run-time errors and also check against standards (MISRA C:2012 used as a guideline)
Dynamic code check	Performed at SIL level
SW complexity	Cyclomatic complexity: 15 - 20 Comments percentages: 20 - 50% Function nesting: 5 - 10 Lines per function: 200 - 300 Lines per file: 1000 - 2000
Code coverage	Lines Coverage: 70 - 80 % Functions Coverage: 70 - 80 % Branches coverage: 70 - 80 %
Requirements tracking	Track SW requirements in the code

5. GNC ASW PROCESS

The GNC ASW process integration starts in the EM-PIL. The code generated during the auto-coding step is integrated into a Linux environment, by the generation of a standalone GNC ASW process. The GNC ASW process will run at 1 Hz and is composed of the GNC ASW code, CSP communication code and auxiliary code needed to integrate and run the visual based navigation. The GNC ASW process is integrated into Juventas onboard computer and will communicate with external processes to receive system information, sensor measurements, images or system FDIR information and to send GNC information and telemetry. The GNC ASW process is depicted in Fig. 8 and follows the EM-PIL approach defined in Fig. 6. The final step is the HIL where the GNC ASW process is integrated into the onboard computer.

Taking advantage of the Linux environment the code coverage activity is performed by using GCOV. This enables accurate tracking of how code was executed and which functions/lines of code have not been reached by the test cases.

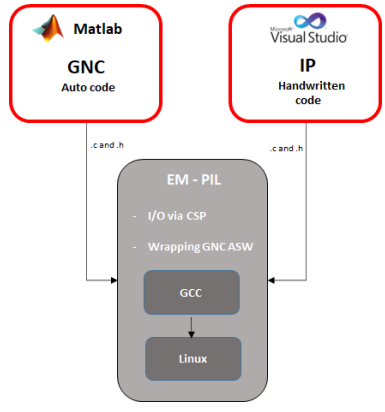

Fig. 8 – GNC ASW process generation

6. CONCLUSIONS

A dedicated visual based GNC SW lifecycle has been proposed to obtain CubeSat flight SW, using a coherent and incremental DDVV strategy. The DDVV is covering the SW from the early stages of design until flight code generation for Juventas CubeSat and concludes with the V&V. The DDVV is developed incrementally and is supported by several test benches: MIL, SIL, EM-PIL and finally HIL. The GNC SW is composed of GNC which is developed as a Simulink model and autogenerated into C code and finally by IP which is handwritten C code. The GNC ASW is finally integrated into a Linux process and later into the final Junventas onboard computer. The DDVV is developed around tailored ECSS-E-ST-40C, ECSS-Q-ST-80C and MISRA C: 2012 standards. A minimum guide list to generate flight code is proposed in chapter 3.

Considering the Juventas experiences a series of remarks can be made:

- The FES design is one of the most challenging phases of DDVV due to multiple activities that are carried out in parallel. To reduce risks, the project schedule has to be detailed with proper time intervals.
- Usually, at the beginning of the project the requirements are not mature enough. To reduce risks, clear functional and SW requirements have to be derived from the beginning of the project
- It is recommended that as soon as the auto-coding step is reached to perform a SIL/EM-PIL to gather preliminary SW information. This recommendation is done to reduce integration risk and to detect issues as soon as possible to reduce design costs.
- It is recommended to use a SIL compiler similar to EM-PIL. In the Juventas case, the GCC compiler is considered for both environments even though versions are different
- The DDVV assumes a continuous iteration between the design team and the SW team.

ACKNOWLEDGEMENT

This article is an extension of the paper presented at *The International Conference of Aerospace Sciences*, "***AEROSPATIAL 2022***", 13 - 14 October 2022, Bucharest, Romania, *Hybrid Conference*, Section 5 – Systems, Subsystems and Control in Aeronautics.

REFERENCES

[1] V. M. Moreno Villa et al., *Mission and GNC system design of the Juventas Cubesat on board the HERA mission*, 11th International ESA Conference on Guidance, Navigation & Control Systems, 22 - 25 June 2021, Virtual.
[2] F. A. Stancu et al., *Validation process from models to HW avionics in the frame of HERA autonomous navigation,* DASIA, 2021.
[3] A. Pellacani et al., *Design, Development, Validation and Verification of GNC technologies,* 8th European Conference for Aeronautics and Sciences (EUCASS), 2019.
[4] * * * SAVOIR: https://savoir.estec.esa.int/SAVOIROrganisation.htm
[5] * * * Simulink - Simulation and Model-Based Design - MATLAB & Simulink (mathworks.com)
[6] * * * Embedded Coder - MATLAB & Simulink (mathworks.com)
[7] F. A. Stancu et al., *Visual based GNC system from prototype to flight software,* Book of Abstracts - International Conference of Aerospace Sciences, "AEROSPATIAL 2022", Hybrid conference, Bucharest, Romania, 13-14 October, 2022, ISSN 2067 – 8614, ISSN-L 2067 – 8614, https://aerospatial-2022.incas.ro/files/book_of_abstracts__aerospatial-2022.pdf