SASlib: An Open Python Library for Galileo Signal Authentication Service (SAS)

Rafael Terris-Gallego*, Oscar Sarda-Marti*, Enrique Takiguchi-Medina*, Aleix Galan-Figueras§, José A. López-Salcedo*, Gonzalo Seco-Granados*, Ignacio Fernandez-Hernandez†

* Univ Autonoma de Barcelona (UAB), CERES, IEEC, Barcelona, Spain
§ KU Leuven, Leuven, Belgium
† DG DEFIS, European Commission, Brussels, Belgium

***Abstract*—The Galileo Signal Authentication Service (SAS) is a new service of the European Galileo Global Navigation Satellite System (GNSS) that provides range protection and complements the navigation data authentication provided by Open Service Navigation Message Authentication (OSNMA). The SAS is based on the encryption of the E6-C signal and the use of the Timed Efficient Stream Loss-tolerant Authentication (TESLA) keys disclosed in the E1-B signal by OSNMA. The service is currently under development, and its initial deployment is foreseen for 2026. This paper presents SASlib, an open Python library implementing the main functions required to process Galileo SAS, including Re-Encrypted Code Sequence (RECS) parsing, snapshot recording, OSNMA TESLA key retrieval, RECS decryption, sample correlation, and signal detection and authentication. The library is intended to support receiver manufacturers and application developers. This paper describes the SASlib architecture and presents initial test results. The reported results provide an initial end-to-end verification with real encrypted E6-C signals.**

***Index Terms*—GNSS, Galileo, SAS, ACAS, OSNMA, TESLA, encryption, authentication, Python, library, open source.**

I. Introduction

Galileo Signal Authentication Service (SAS), previously known as Assisted Commercial Authentication Service (ACAS), is the latest service that the Galileo system is developing to combat the increasing threat of spoofing against Global Navigation Satellite System (GNSS) [1]. By leveraging the existing structure of the Galileo signals, the SAS provides signal authentication at the range level. This service, currently in the definition and development phase, is expected to be initially deployed in 2026 [2]. It complements the Open Service Navigation Message Authentication (OSNMA) [3] service, which already provides navigation data authentication. Together, both services will provide robust protection against spoofing attacks from a system-level perspective.

Galileo SAS is the first public system aimed at providing range authentication, although GPS is also expected to offer similar protection with Chips-Message Robust Authentication (CHIMERA) [4]. Some commercial systems have also reported range authentication capabilities [5].

Galileo SAS is based on the encryption of the E6-C signal and the use of the Timed Efficient Stream Loss-tolerant Authentication (TESLA) keys disclosed in the E1-B signal by OSNMA. Selected fragments of the encrypted data in E6-C, known as Encrypted Code Sequences (ECSs), are re-encrypted at the ground segment prior to their transmission, using the OSNMA keys that are yet to be disclosed. The resulting Re-Encrypted Code Sequences (RECSs) are then made available to users through a publicly accessible repository, so that they can be downloaded in advance by any SAS-ready receiver that wishes to authenticate its position at the epochs of interest. Once the E6-C signal is broadcast, the receiver records a snapshot of E6-C around the epoch when the RECSs are expected. After the corresponding TESLA keys are disclosed in the E1-B signal, the receiver can decrypt the RECSs and then correlate them with the recorded E6-C samples to detect the presence of the signal and authenticate the receiver Position, Velocity and Timing (PVT) if certain conditions are met [6].

Preliminary studies and simulations have been carried out to define the main parameters of the SAS and to evaluate its theoretical performance under different scenarios, including the desired length and periodicity of the RECS [7] [8]. These studies have been complemented with experimental measurements using Software Defined Radio (SDR)-based platforms [9] and the unencrypted E6-C signal [10], which is expected to be permanently encrypted soon. Recently, a test campaign was carried out by the European Commission and ESA, in which the eccentric E14 satellite transmitted the E6-C signal encrypted for a short period. This campaign made it possible to test the SAS concept in a real scenario and to validate end-to-end the SAS processing chain, including both system and

This work was supported in part by the European Commission Defence Industry and Space Satellite Navigation (DEFIS) contract DEFIS/2023/OP/0011, in part by the Spanish Agency of Research (AEI) under grant PID2023-152820OB-I00 funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU, and in part by the Catalan ICREA Academia Programme. The content of this article does not necessarily reflect the official position the authors' organizations. Responsibility for the information and views set out in this article lies entirely with the authors.

979-8-3503-5685-4/24/$31.00 ©2026 IEEE

receiver operations. The results obtained in this campaign were in line with expectations and represented a significant step forward in Galileo SAS development and its future initial deployment for the testing phase [11] [12].

To support receiver manufacturers and application developers implementing Galileo SAS, we developed SASlib, an open-source Python library that implements the main functions required for its processing. It is publicly available on GitHub [13], where users can access it, use it, and contribute to its development, in line with the OSNMAlib library published to support OSNMA [14] [15].

The scope of this paper is the presentation of the SASlib architecture and its end-to-end verification with real encrypted E6-C signals, rather than a complete performance characterization campaign.

This paper is organized as follows. Section II describes the architecture of the SASlib library and its main modules. Section III explains the library functionality and execution process, including its data formats, graphical interface, availability, and documentation. Section IV presents the initial test results obtained with the library using real encrypted E6-C test-campaign data. Finally, Section V summarizes the main conclusions of this work.

II. SASlib Architecture

This section outlines the SASlib architecture and its main modules. The library is modular and extensible, so modules can be executed independently and combined according to user needs. For testing, modules that require input data can use previously downloaded or recorded local files, or acquire the data live as in normal operation. The main modules of the library, which are illustrated in Figure 1, are described below.

A. Startup modules

These modules handle the initial steps needed to prepare the receiver for authentication. They include the Configuration Initialization, Time Reference Synchronization, Almanac Management, and RECS/BGD/SLOG Download and Parsing modules. They are typically executed before signal broadcast, when a broadband Internet connection is available to retrieve the required files.

Configuration Initialization: This module initializes the configuration parameters required by the other modules and defines the main user settings for Galileo SAS processing, such as the authentication start time, duration and periodicity (RECS Period), and the auxiliary signal band used to assist RECS detection. These parameters are sufficient to run SASlib, but additional settings, such as the RECS length, the randomization parameters, or data source, can also be configured. For testing purposes, the library can be configured either to use local files for all required data or to download/record them as needed.

Time Reference Synchronization: This module provides a time reference, together with its associated uncertainty bound, to be used for snapshot acquisition. A larger uncertainty bound requires a longer snapshot to ensure that the RECS are captured within the recorded samples. Currently, the time reference is obtained from an Network Time Security (NTS) server. Future releases may include additional time sources, such as GNSS-based solutions.

Almanac Management: This module provides *assistance data* for the receiver by downloading the Galileo almanac XML file published by the European GNSS Service Centre (GSC) (https://gsc-europa.eu). Although it is not required for SAS processing, it can be used as optional assistance to speed up signal acquisition in the subsequent Signal Correlation module. In particular, for each satellite, the downloaded XML file is parsed to extract the six Keplerian orbital elements ($\sqrt{a}$, eccentricity, inclination, RAAN, argument of perigee, and mean anomaly), together with the reference epoch (time of applicability and week number) and the health status. These parameters are used to estimate satellite visibility and expected Doppler/code-delay values, thereby reducing the search space. For testing purposes, this module can also be configured to load the almanac from a local file.

RECS/BGD/SLOG Download: This module downloads the RECS/BGD/SLOG files from a user-configured publicly accessible server. The files are downloaded in advance to ensure the required level of autonomy at the time of authentication. Once Galileo SAS becomes operational, the intended source is the GSC, which is expected to act as the official server, although it is not yet operational. As specified in the latest specification available [1], the module queries the selected server using the user-configured parameters and retrieves a TAR file containing all the required files.

RECS/BGD/SLOG Parsing: This module extracts the binary RECS files from the aggregated files contained in the downloaded TAR archives, following the latest SAS specification [1]. It then parses them to recover the RECS bits. It also processes the Broadcast Group Delay (BGD) files to obtain the corresponding BGD values, and checks the SLOG files to verify the SAS status.

B. Snapshot modules

These modules record E6-C signal snapshots around the expected reception time of the RECSs. Unlike the startup modules, they are executed close to the authentication epoch and are designed to operate with limited Internet connectivity. This group comprises the Uncertainty Estimation and Snapshot Recording modules.

Uncertainty Estimation: Because time may have elapsed since the time reference was obtained in the Time Reference Synchronization module (typically due to receiver clock drift), the previous reference may have a

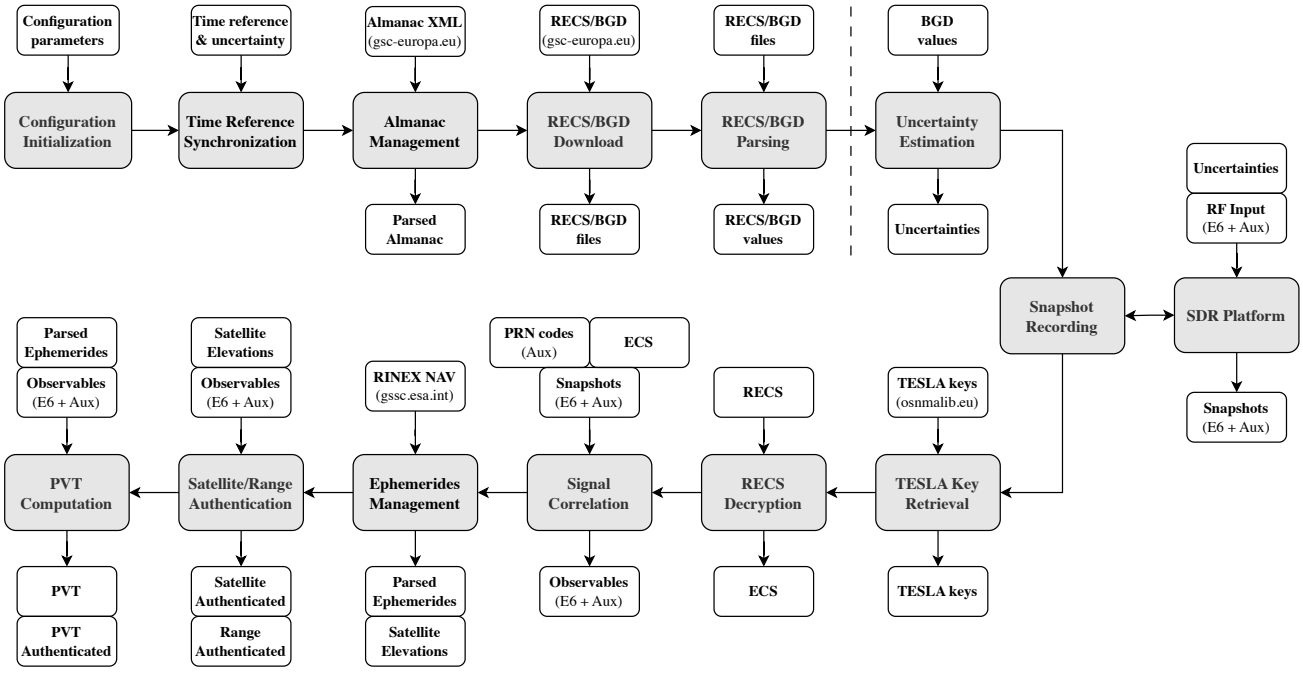

Fig. 1. SASlib architecture and main functions.

higher uncertainty and should be updated (when possible), and a tighter uncertainty bound should be estimated closer to the recording time. If an Internet connection is available, this module reconnects to an NTS server to update the time reference. In addition to the uncertainty of this time reference (which determines the uncertainty of the receiver clock), this module also configures the additional delay contributions considered for snapshot acquisition: the signal propagation delay uncertainty (up to 20 ms in Galileo, corresponding to the difference in propagation time between the closest and farthest satellites), the satellite clock offset (typically set to 0 ms when BGDs are applied), and the SAS specific parameters that affect the snapshot length, such as the RECS length and the randomization parameters (RAND). The total uncertainty bound is obtained by summing all these contributions and is then used to determine the required snapshot length.

Snapshot Recording: This module commands the recording of an E6 signal snapshot around the expected time of reception of the RECS. It can also be configured to command the recording of a snapshot from an auxiliary band to assist subsequent E6-C signal acquisition. This module currently interfaces, through a dedicated Application Programming Interface (API), with a remotely accessible SDR-based platform at the Univ. Autonoma de Barcelona (UAB). The platform is permanently connected to a multi-band GNSS antenna and includes two bladeRF boards for synchronous sample acquisition, as shown in Figure 2 [9]. When an external recording platform is used, the receiver clock uncertainty from the Uncertainty Estimation module is disabled, since the relevant clock

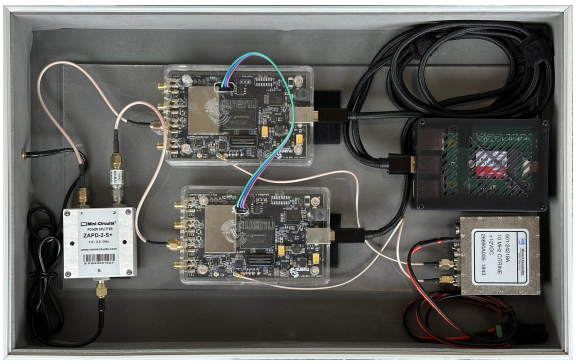

Fig. 2. SDR-based platform for synchronous sample acquisition.

uncertainty is that of the recording platform. In this case, only the satellite clock offset and signal propagation delay terms are sent through the API. The external platform then combines them with its own estimated clock uncertainty to determine the required snapshot length. Alternatively, this module can be configured to load previously recorded snapshots from local files for post-processing.

C. ECS Generation modules

These modules generate the ECSs, i.e., the local replicas used for correlation, after the E6-C signal has been recorded and the corresponding TESLA keys have been disclosed. They comprise the TESLA Key Retrieval and RECS Decryption modules.

TESLA Key Retrieval: Once the snapshots are recorded, SASlib waits until the corresponding TESLA keys are disclosed in the Galileo E1-B signal. This module

retrieves those keys from OSNMA. The keys are currently obtained from an OSNMAlib server (osnmalib.eu). In addition, for testing purposes, this module can be configured to load the keys from a local file.

RECS Decryption: This module uses the retrieved OSNMA keys to decrypt the RECS and obtain the corresponding ECSs. These ECSs are local replicas that are later correlated with the previously recorded E6 snapshot samples. The decryption process follows the specifications defined in the latest SAS Interface Control Document (ICD) [1], which is currently based on Advanced Encryption Standard (AES)-Cipher Block Chaining (CBC) mode.

D. Signal Processing and Authentication modules

These modules perform the main signal processing and authentication steps once the ECS has been generated from Signal-In-Space (SIS). They include Signal Correlation, Ephemerides Management, Satellite/Range Authentication, and PVT Computation.

Signal Correlation: This module correlates the recorded E6-C samples with the ECSs to obtain the E6-C observables, including code delay, Doppler frequency, and the carrier-to-noise density ratio (C/N_0). If an auxiliary-band snapshot has also been recorded, the module first correlates the auxiliary-band samples with the corresponding local replicas (PRN codes) to obtain the auxiliary-band observables. These observables are then used to estimate the expected E6-C code delay and Doppler frequency, helping to narrow the E6-C correlation search window. In practice, when an auxiliary signal is used, only a few time- and frequency-domain correlations may be sufficient. This approach significantly reduces the computational load of the correlation process, as described in [7].

Ephemerides Management: This module retrieves Galileo broadcast ephemerides (RINEX NAV) from a public server, such as the ESA GNSS Science Support Centre (GSSC) [16], which provides hourly updates via SFTP, and parses them for PVT computation. Since ephemerides have a limited validity (hours) and several days may elapse between initialization and the authentication epoch, they are downloaded close to the PVT stage to avoid outdated data. The module also computes satellite elevations used by the Galileo SAS authentication mechanism. For testing, it can alternatively load ephemerides from a local file.

Range Authentication: Using the E6-C observables obtained in the previous module, this module explicitly checks whether the E6-C measurements can be considered as authentic (i.e., RECS has been successfully detected). This can be done either by applying Primary Peak to Secondary Peak (PPSP) or by comparing the E6-C correlation peak value against the noise floor. This step is referred to as *range authentication*. Additionally, the auxiliary-signal range is authenticated by comparing the code-delay difference between the auxiliary signal and E6-C. If this difference is within a given threshold, the auxiliary-signal range is considered authentic, as specified in [6] and based on the authentication mechanism described in [17]. This step, referred to as *auxiliary-range authentication*, uses satellite elevations to compute the expected code-delay difference and to set the corresponding threshold.

PVT Computation: This module computes the receiver PVT using observables obtained from the Signal Correlation module. By default, the PVT solution is computed from the auxiliary-signal observables. However, if four or more ranges from different satellites are authenticated by the Satellite/Range Authentication module, the PVT is considered authenticated at the position level. At the time of writing, only two satellites transmit the E6-C signal encrypted (E14 and E18), so position-level authentication is not yet possible. Nevertheless, this functionality is included in the library for future use as more satellites begin transmitting encrypted E6-C.

III. Functionality & Execution

This section outlines how SASlib modules are configured and executed to build processing pipelines.

A. Data Format

SASlib adopts a JSON-based configuration architecture designed to ensure modularity, traceability, and experimental reproducibility. This approach decouples parameter specification from algorithmic implementation through a structured data management strategy.

Each processing block is defined by a JSON Schema file specifying its configurable parameters, including data types, valid ranges, default values, and descriptive metadata. These schemas enable automatic parameter validation and allow the graphical interface to generate the corresponding input forms dynamically.

The framework supports two complementary configuration approaches. Users may define a single global JSON file containing the parameters of all modules together with execution metadata such as timestamps, identifiers, and version information. When loaded, the system distributes its contents into the corresponding block-specific JSON configuration files. Alternatively, users may edit the block-specific files directly to adjust individual modules. Both approaches are interoperable: a global configuration can be expanded into block-specific files, and block-specific files can be consolidated into a single global configuration for archival or sharing.

Upon execution, all parameters and outputs are stored in timestamped directories, preserving the complete state of each run for traceability and recovery.

B. Graphical Interface

To facilitate the use of the library, a graphical interface has also been developed. It allows users to configure and run the different modules and visualize the results. The SASlib graphical interface is built on Dash and uses

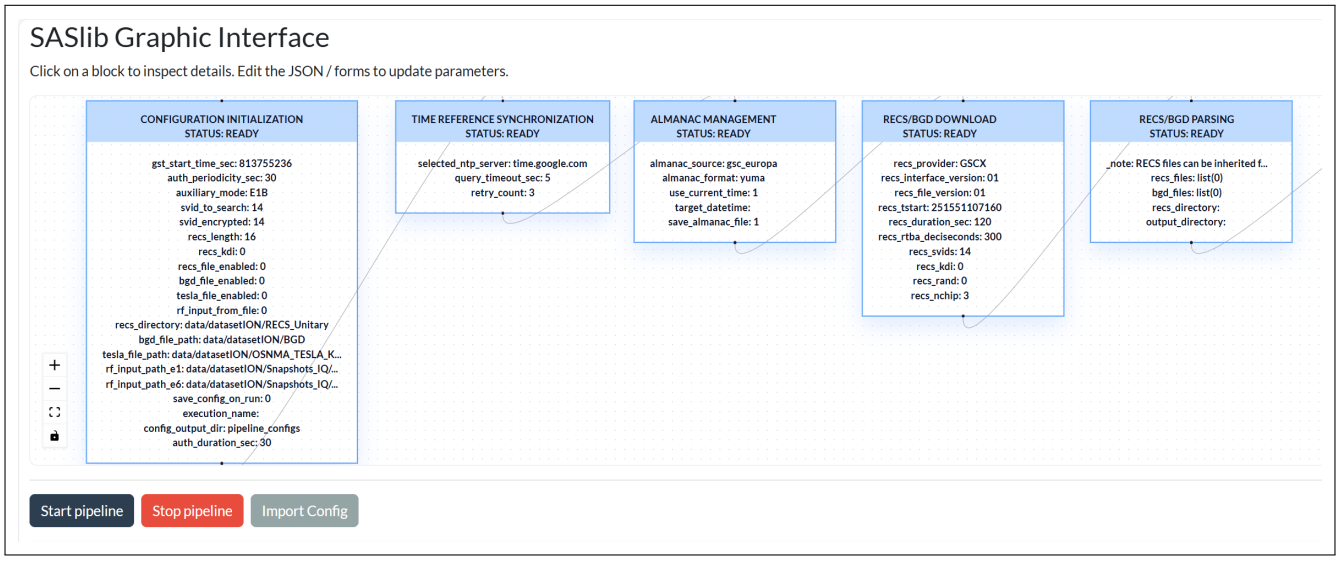

Fig. 3. SASlib graphical interface.

ReactFlow to represent the pipeline as connected nodes. Each node is defined by a specific JSON file and linked at runtime to its backend module function. Selecting a node opens a modal with editable parameters, execution controls, and relevant logs.

The interface supports running single nodes or full pipelines, and all configuration files are saved to ensure reproducibility. Figure 3 shows a screenshot of the graphical interface, and Figure 4 shows the Configuration Initialization module.

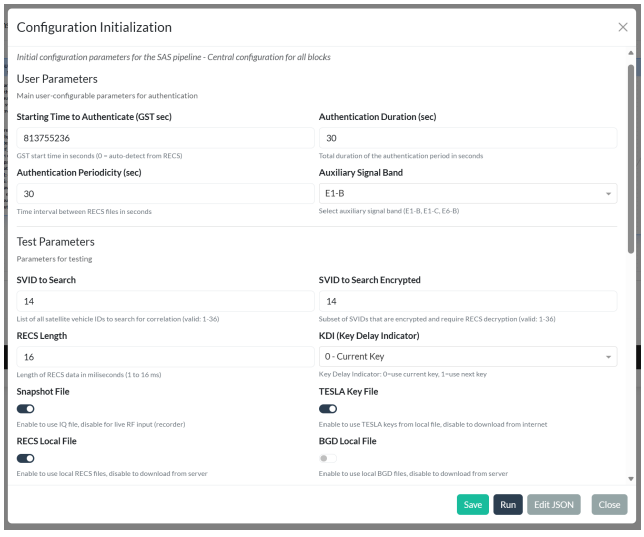

Fig. 4. SASlib Configuration Initialization module.

C. Availability & Documentation

Galileo SASlib is publicly available on GitHub as an open-source Python library [13]. The repository includes the source code, configuration schemas, example datasets for testing and validation, and a README with installation and usage instructions.

Regarding code quality, the codebase follows PEP 8 style conventions, uses PEP 484 type annotations throughout to support static analysis and improve readability, and documents functions using Google-style docstrings.

SASlib is released under the European Union Public Licence (EUPL), Version 1.2 (EUPL), the official license of the European Union. As defined at [18], the license is interoperable, reciprocal, share-alike, and compatible.

IV. Results

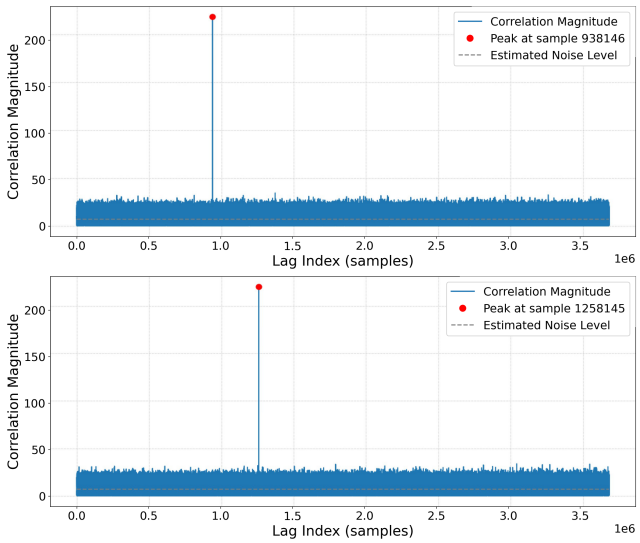

Fig. 5. Results obtained with SASlib using the encrypted E6-C test dataset (KDI=0 on top, KDI=1 on bottom).

This section verifies the end-to-end operation of SASlib using real encrypted E6-C signals. The results are therefore presented as a functional validation of the processing

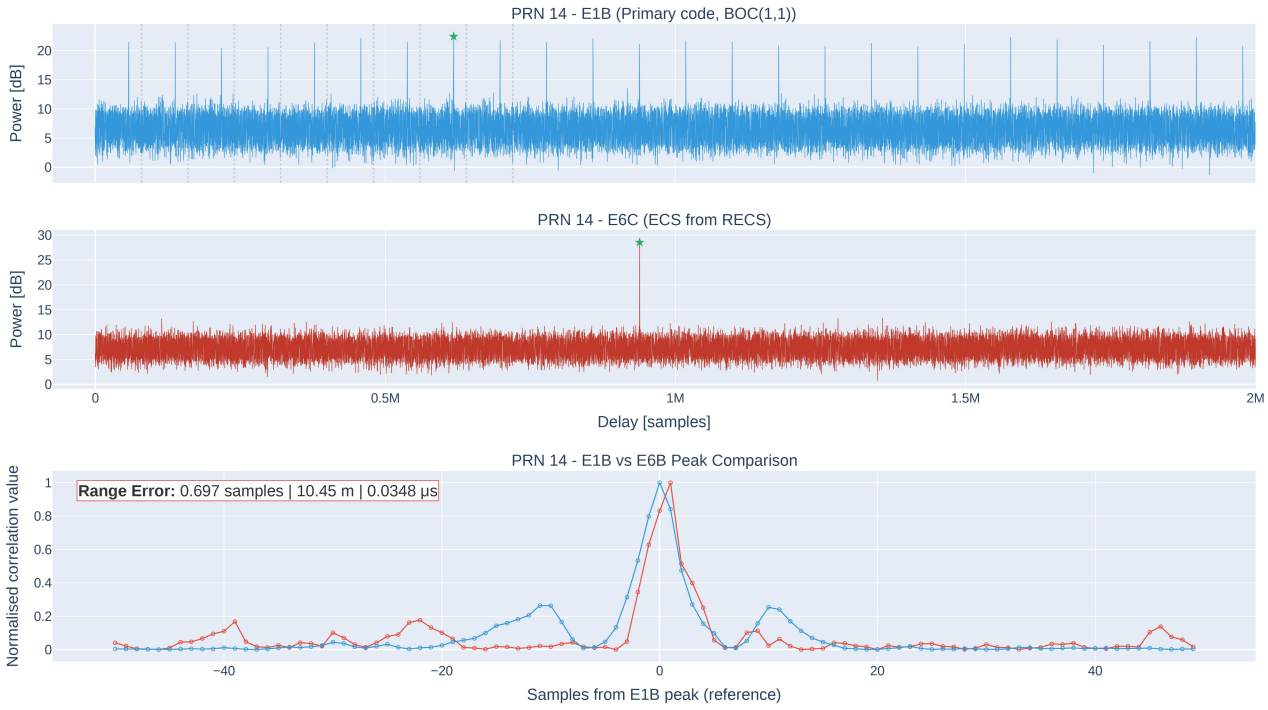

Fig. 6. Results obtained with SASlib using a test dataset with an encrypted E6-C signal. From top to bottom: E1-B samples correlation, E6-C samples correlation, and comparison of E1-B correlation peak (blue) and E6-C correlation peak (red).

chain rather than as a complete statistical characterization of SAS performance. SASlib was tested with real data collected during the Galileo SAS campaign carried out in June 2025, in which the eccentric-orbit E14 satellite transmitted the E6-C signal encrypted for a short period [11] [12]. The same datasets used in [11] were processed with SASlib and compared against the reference-platform results obtained during the campaign.

The results obtained with SASlib are consistent with those obtained with the reference platforms. Figure 5 shows the E6-C correlation peaks for both Key Delay Index (KDI) configurations. These peaks are obtained by correlating the dataset samples with the ECSs obtained after decrypting the RECSs using the corresponding TESLA keys. The peaks appear at the expected code delays and Doppler frequencies, and the separation between the two peaks is consistent with the configured KDI offset, which is 16 ms as specified in [1]. At the sampling rate used (20 Msps), this corresponds to 320 k samples.

In Figure 6, the acquisition outputs produced by the Signal Correlation module for the E1-B and E6-C signals are shown. These results are compared with the E6-C peak (for the KDI=0 case) to verify the alignment of the code delays, which is a crucial aspect of the Galileo SAS authentication mechanism. A zoomed view around the E6 peak is also provided to better illustrate the code-delay alignment. The resulting code-delay difference is then passed to the Range Authentication module, where it is compared against a threshold to determine whether the range can be authenticated, as specified in [6].

V. Conclusion

The main contribution of this paper is the development of SASlib, an open-source, modular Python library that implements the end-to-end receiver processing chain for Galileo SAS, including integration with an SDR-based, multi-band, synchronous snapshot platform for remote Galileo E6/E1 sample recording. The library has been validated experimentally using real encrypted E6-C test-campaign data, demonstrating correlation and detection performance consistent with reference platforms. It is publicly available on GitHub to support adoption by receiver manufacturers and developers and to facilitate further work on Galileo SAS.

Further testing and validation are planned under a wider range of scenarios and conditions, including a more complete performance characterization with real signals. The developers also plan to add new functionality, including support for additional authentication mechanisms such as Vestigial Signal Search (VSS), and to foster collaboration in support of the evolution of Galileo SAS.

References

[1] I. Fernandez-Hernandez, J. Winkel, C. O'Driscoll, G. Caparra, R. Terris-Gallego, et al., "Galileo Signal Authentication Service (SAS)," in *Proceedings of the 37th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+*

2024), Baltimore, Maryland, USA: Institute of Navigation, Sep. 2024, pp. 3292 –3307. DOI: 10.33012/2024.19707

[2] I. Fernandez-Hernandez, J. Winkel, T. Willems, S. Cancela, M. Ramírez, et al., "Prototyping Galileo Signal Authentication Service: Current Status and Plans," in *Proceedings of the European Navigation Conference (ENC 2025)*, Wroclaw, Poland: MDPI, May 2025. DOI: 10.3390/engproc2026126040

[3] European Union, "Galileo Open Service Navigation Message Authentication (OSNMA) Signal-In-Space Interface Control Document (SIS ICD), Issue 1.1," Tech. Rep., Oct. 2023. DOI: 10.2878/594840

[4] J. M. Anderson, K. L. Carroll, N. P. DeVilbiss, J. T. Gillis, J. C. Hinks, et al., "Chips-Message Robust Authentication (Chimera) for GPS Civilian Signals," in *Proceedings of the 30th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2017)*, Portland, Oregon, USA: Institute of Navigation, Sep. 2017, pp. 2388–2416. DOI: 10.33012/2017.15206

[5] J. Anderson, "World's First Authenticated Satellite Pseudorange from Orbit," in *Proceedings of the 38th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2025)*, Baltimore, Maryland, USA: Institute of Navigation, Sep. 2025. DOI: 10.33012/2025.20365

[6] I. Fernandez-Hernandez, J. Winkel, C. O'Driscoll, S. Cancela, R. Terris-Gallego, et al., "Semiassisted Signal Authentication for Galileo: Proof of Concept and Results," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 4, pp. 4393–4404, Feb. 2023. DOI: 10.1109/TAES.2023.3243587

[7] R. Terris-Gallego, I. Fernandez-Hernandez, J. A. López-Salcedo, and G. Seco-Granados, "Guidelines for Galileo Assisted Commercial Authentication Service Implementation," in *Proceedings of the 12th International Conference on Localization and GNSS (ICL GNSS 2022)*, Tampere, Finland: IEEE, Jun. 2022. DOI: 10.1109/ICL-GNSS54081.2022.9797027

[8] R. Terris-Gallego, J. A. López-Salcedo, G. Seco-Granados, and I. Fernandez-Hernandez, "Operating Modes and Performance Evaluation of Galileo Assisted Commercial Authentication Service," in *Proceedings of the 35th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2022)*, Denver, Colorado, USA: Institute of Navigation, Sep. 2022. DOI: 10.33012/2022.18441

[9] R. Terris-Gallego, J. A. López-Salcedo, G. Seco-Granados, and I. Fernandez-Hernandez, "E1-E6 SDR platform based on bladeRF for testing Galileo Assisted Commercial Authentication Service," in *Proceedings of the European Navigation Conference (ENC 2023)*, ESA ESTEC, Noordwijk, The Netherlands: MDPI, Oct. 2023. DOI: 10.3390/ENC2023-15428

[10] R. Terris-Gallego, J. A. López-Salcedo, G. Seco-Granados, and I. Fernandez-Hernandez, "Preliminary Evaluation of Galileo ACAS using Existing E1-E6 Open Signals and a Low-Cost SDR Platform," in *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*, Denver, Colorado, USA: Interna- tional Conference on Localization and GNSSInstitute of Navigation, Sep. 2023, pp. 1388–1402. DOI: 10.33012/2023.19319

[11] R. Terris-Gallego, J. A. López-Salcedo, and G. Seco-Granados, "Testing Galileo SAS with Encrypted Signal-in-Space," in *Proceedings of the 38th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2025)*, Baltimore, Maryland, USA: Institute of Navigation, Sep. 2025. DOI: doi.org/10.33012/2025.20223

[12] M. A. Ramirez, S. Cancela, D. Calle, I. Fernandez-Hernandez, T. Willems, et al., "Looking into the Galileo Signal Authentication Service: Early demonstration of the service and experimentation," in *Proceedings of the 38th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2025)*, Baltimore, Maryland, USA: Institute of Navigation, Sep. 2025. DOI: 10.33012/2025.20469

[13] SPCOMNAV (Univ. Autonoma de Barcelona), *Galileo SASlib GitHub repository*, Apr. 2026. [Online]. Available: https://github.com/SPCOMNAV/GalileoSASlib

[14] A. Galan-Figueras, I. Fernandez-Hernandez, L. Cucchi, and G. Seco-Granados, "OSNMAlib: An Open Python Library for Galileo OSNMA," in *Proceedings of the 10th Workshop on Satellite Navigation Technology (NAVITEC 2022)*, ESA ESTEC, Noordwijk, The Netherlands: IEEE, Apr. 2022, pp. 1–12. DOI: 10.1109/NAVITEC53682.2022.9847548

[15] A. Galan-Figueras, C. Iñiguez, I. Fernandez-Hernandez, S. Pollin, and G. Seco-Granados, "Improving OSNMAlib: New Formats, Features, and Monitoring Capabilities," *IEEE Journal of Indoor and Seamless Positioning and Navigation*, vol. 3, pp. 117–127, Apr. 2025. DOI: 10.1109/jispin.2025.3558771

[16] ESA, *ESA GNSS Science Support Centre*. [Online]. Available: https://gssc.esa.int

[17] F. Ardizzon, G. Caparra, I. Fernandez-Hernandez, and C. O'Driscoll, "A Blueprint for Multi-Frequency and Multi-Constellation PVT Assurance," in *Proceedings of the10th Workshop on Satellite Navigation Technologies (NAVITEC 2022)*, ESA ESTEC, Noordwijk, The Netherlands, Apr. 2022.

[18] European Union, *European Union Public Licence v.1.2*, 2016. [Online]. Available: https://eupl.eu/1.2/en/