Low-cost SDR platforms for multi-frequency GNSS synchronous acquisition

Rafael Terris-Gallego*, Aleix Galan-Figueras§, Ignacio Fernandez-Hernandez†,
José A. López-Salcedo*, Gonzalo Seco-Granados*

* Universitat Autònoma de Barcelona (UAB), IEEC-CERES, Barcelona, Spain
§ Katholieke Universiteit Leuven (KUL), Leuven, Belgium
† DG DEFIS, European Commission, Brussels, Belgium

2024 11th Workshop on Satellite Navigation Technology (NAVITEC) | 979-8-3503-5685-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/NAVITEC63575.2024.10843629

***Abstract*—The use of multiple frequencies in GNSS allows for greater accuracy of the navigation receiver. This not only reduces the impact of the ionosphere, but also provides additional signals to enhance robustness. In various applications, receivers require synchronous access to samples from different frequency bands. In this paper, we provide an overview of several low-cost SDR platforms that enable synchronized sample capture and describe how to configure them for this purpose. Finally, we discuss GNSS applications, such as Galileo SAS, that can take advantage of these platforms for testing and evaluation.**

***Index Terms*—GNSS, SDR, bladeRF, HackRF, multi-frequency, synchronization, acquisition, ACAS, SAS.**

I. Introduction

Global Navigation Satellite System (GNSS) are essential for providing accurate Position, navigation and timing (PNT) services across various markets. Using multiple frequencies in GNSS greatly enhances performance by improving data accuracy and reliability. This approach helps mitigate errors caused by ionospheric delays or multipath effects, leading to more precise positioning.

Furthermore, GNSS services like Galileo Signal Authentication Service (SAS) [1]—formerly known as Assisted Commercial Authentication Service (ACAS)—also leverage the use of multiple frequency bands. SAS aims to provide authentication of the ranging signal through the encryption of the E6-C signal and the use of the Timed Efficient Stream Loss-tolerant Authentication (TESLA) keys broadcast in the E1-B signal by Open Service Navigation Message Authentication (OSNMA).

The Position, Velocity and Timing (PVT) authentication mechanism compares the code delay estimates from E1-B and E6-C. If the difference is below a predefined threshold, the signal can be authenticated [2]. Thus, perfectly aligned samples from both frequency bands are critical for SAS operation. Therefore, affordable platforms capable of synchronous sample capture at different frequencies are essential for testing and evaluating these services.

This work was supported by the Spanish Agency of Research project PID2023-152820OB-I00 and by the Catalan ICREA Academia Program, and partially funded by the Research Foundation Flanders (FWO) project number 1SH9424N. The content of this article does not necessarily reflect the official position of the authors' organizations. Responsibility for the information and views set out in this article lies entirely with the authors.

The adoption of Software Defined Radio (SDR) boards has become popular due to their unparalleled flexibility, affordability, and ease of experimentation. SDR-based platforms allow for rapid adaptation and system modification without physical changes, thus accelerating the development cycle. In [3], an SDR experimental platform using low-cost bladeRF micro 2.0 boards was presented. This platform provided a preliminary evaluation of SAS using existing Galileo open signals [4]. However, it is not the only low-cost option for synchronizing two boards; HackRF One models also support synchronized capture of multiple frequency bands at an even lower cost [5].

In this paper, we compare the two boards and describe the configuration process for synchronized capture of multiple frequencies. We detail the specific connections and commands required so that any researcher or developer can replicate the setup. Additionally, we highlight critical considerations, such as potential sample overruns at high sampling rates. Finally, we present the results from correlating the samples and discuss their implications for specific GNSS applications.

II. SDR platforms for synchronous acquisition

A. SDR platforms description

The bladeRF 2.0 micro board (see Figure 1) includes a half-duplex AD9361 transceiver from Analog Devices. The front-end operates at frequencies ranging from 47 MHz to 6 GHz, with an IQ sampling rate of up to 60 Msps at a resolution of 12 bits. This rate can be extended to 120 Msps using the 8-bit mode support of the transceiver [6]. The list of specifications can be found in [7]. Although the bladeRF micro 2.0 board offers 2×2 MIMO capabilities, both transmitters and receivers share the same oscillator. This prevents acquiring synchronous samples of bands located far apart, requiring the use of two separate boards to achieve this.

The HackRF One board also includes a half-duplex transceiver but uses the simpler MAX2837/MAX2839 versions from Analog Devices. The front-end operates at frequencies from 1 MHz to 6 GHz, with an IQ sampling rate of up to 20 Msps at a resolution of 8 bits.

Similar to the bladeRF board from Nuand, the HackRF One (see Figure 1) is well supported in the SDR community due to its simplicity, long-term support, open-source hardware, and low entry price. The list of specifications can be found in [8].

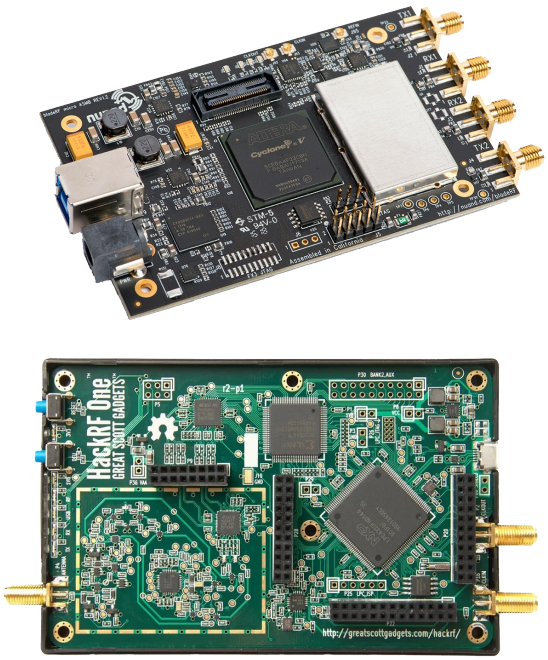

Fig. 1. bladeRF micro 2.0 (top) and HackRF One (bottom) boards.

Although the device was cutting-edge when it was released, it is now starting to fall behind competitors due to its limited sampling rate of 20 Msps, the use of High Speed USB (USB 2) instead of Super Speed USB (USB 3), and a relatively unstable clock board that generates the reference signals. For GNSS purposes, it is highly recommended to connect an external Temperature-Compensated Crystal Oscillator (TCXO) into the appropriate headers to provide a more stable reference, or to use an external clocking source through the CLKIN SMA port.

Even with a stable reference clock, the HackRF's fractional-N Phase-Lock-Loops (PLLs) in the radio-frequency chain may not be able to generate the exact frequency requested by the user. This is particularly impactful when working with GNSS, as tuning the HackRF to the E1 center frequency of 1575.42 MHz results in a 21 Hz offset using the latest firmware. Although this is a small difference, it has significant implications when trying to track the phase of the signal [9]. In contrast, the bladeRF board shows a significantly smaller offset, measured at only 2 Hz.

Regardless of that, to focus on the synchronization capabilities of the SDR boards, an external reference has been used in both cases, ensuring good short-term stability.

In Table I the maximum values that can be achieved for the main parameters of both SDR boards are summarized.

Board Model	Freq.	IQ Rate	ADC	USB	Price
bladeRF µ2.0	6 GHz	60 Msps	12 bits	USB 3	500 €
HaceRF One	6 GHz	20 Msps	8 bits	USB 2	300 €

TABLE I
SDR BOARDS MAIN SPECIFICATIONS (MAXIMUM VALUES).

B. BladeRF synchronization procedure

The synchronization procedure for the bladeRF micro 2.0 is based on using hardware triggering. This procedure was not officially documented when the first experimental platform was implemented [3], but has since been updated with all the required steps to achieve it [10].

The two bladeRF boards are connected to the same multi-band antenna using a splitter, with the RX SMA connectors. Furthermore, both boards are connected to an external clock via the REFIN J95 UFL connector. Sharing the same external clock prevents any mismatches between the internal clocks of both boards. For this setup, we used an Oven-Controlled Crystal Oscillator (OCXO), which provides a very stable reference. This setup is illustrated in Figure 2. By default, the

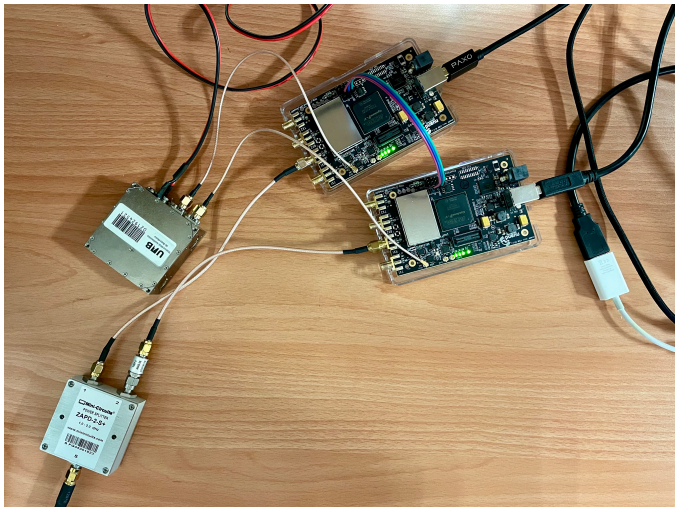

Fig. 2. bladeRF synchronization setup.

bladeRF boards use their internal clocks. To enable the use of the external reference, the command '`set clock_ref enable`' is used. If needed, the frequency of the clock reference can be configured with the command '`set refin_freq 10M`'. In this case, a 10 MHz frequency is configured.

The boards are set to record samples of the Galileo E1 and E6 frequency bands. Therefore, the front-end frequencies are configured with the commands '`set frequency rx 1575.42M`' for E1 and '`set frequency rx 1278.75M`' for E6. The sampling rate is configured with the command '`set samplerate 10M`' (i.e., 10 Msps).

Next, the test points for triggering are defined. First on the master board, using the commands provided below:

```
rx config file=master_e6.bin n=100M timeout=10s
trigger j51-1 tx master; trigger j51-1 rx slave
rx start
```

And then on the slave board, with the following commands:

```
rx config file=slave_e1.bin n=100M timeout=10s
trigger j51-1 rx slave
rx start; rx wait
```

In addition to the filename and the total number of samples to be recorded (‘`n = 100 M`’), the ‘`rx config`’ command allows specifying a timeout to prevent an error from being reported if the trigger on the master board is executed more than one second later than the slave receiver, which is the default time if this option is omitted.

Finally, we need to fire the trigger back into the master board to start acquiring samples synchronously, which is done with the command ‘`trigger j51-1 tx fire`’.

It is important to note that overruns may occur if using a sampling rate approaching the limit of the board. The following command allows checking for such overruns:

```
bladeRF-cli -d "*:serial=$MASTER" -v debug -s rx_master.
    bladerf
```

The synchronization procedure is summarized next:

1) Connect external clock via REFIN J95 UFL connector.
2) Enable external clock with ‘`set clock_ref enable`’.
3) Connect the trigger output from the master board to the trigger input on the slave board (J51 test points).
4) Arm the triggers in both the master and slave boards.
5) Fire the trigger from the master board.

C. HackRF synchronization procedure

The synchronization procedure for the HackRF is also based on the so-called hardware triggering, and has been available in the firmware code for several years [5]. However, some versions of the firmware had a bug that affected this feature. This was the case with firmware 2023.01.1, which was released specifically for the new hardware revision r9 (the first that used the MAX2839 transceiver), where the triggering feature did not work unless the board used was an r9. In the latest firmware release at the time of writing this manuscript (firmware 2024.02.1), this bug has been resolved, and the hardware triggering function works on any hardware revision.

To configure the boards for hardware triggering, they must first share a clocking signal. This can be achieved by connecting an external clock to the clock-in port of both HackRF boards or by connecting the clock-out of one board to the clock-in of the other. For the latter scenario, clock-out needs to be enabled with the command ‘`hackrf_clock -d $MASTER -o 1`’. If a TCXO is used, this reference is the one shared through the clock-out port instead of the internal clock. By default, HackRF boards use an external clock signal if available, which can be verified with ‘`hackrf_clock -d $BOARD -i`’.

The next step for synchronization is to share a common ground and connect the trigger output pin of the master board to the trigger input pin of the slave board. If one board is sharing the clock with the other, they already have a common ground; alternatively, we connect the P28 pin 2 of both boards using a wire. For the trigger signal, we connect the P28 pin 15 of the master board to the P28 pin 16 of the slave board [11], as shown in Figure 3.

To record samples synchronously on both boards, the master needs to be configured with the following commands:

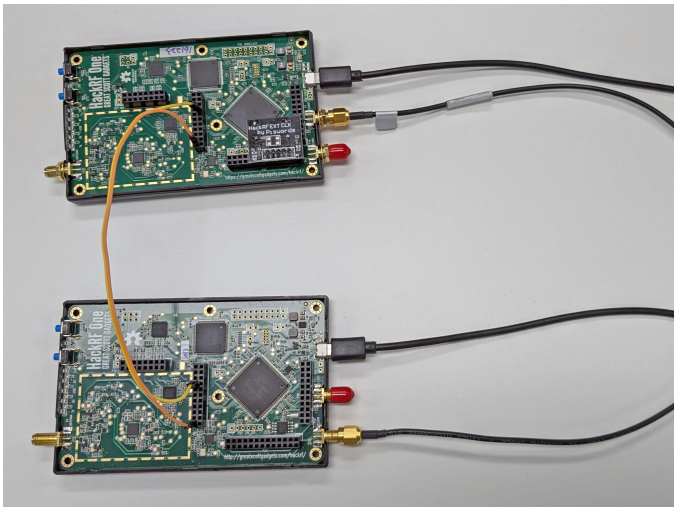

Fig. 3. HackRF synchronization setup.

```
hackrf_transfer -d $MASTER -r hackrf_E1_10Msps_gain.bin -f
     1575420000 -s 10000000 -B -n 200000000
```

The slave board needs to be configured as follows:

```
hackrf_transfer -H -d $SLAVE -r hackrf_E6_10Msps_gain.bin
    -f 1278750000 -s 10000000 -B -n 200000000
```

Here, ‘`-d`’ specifies the board, ‘`-r`’ names the file, ‘`-f`’ sets the center frequency, ‘`-s`’ defines the sample rate, and ‘`-n`’ indicates the number of samples to record. The ‘`-H`’ option on the slave board arms the trigger and waits for the master board’s signal to start recording.

It is important to highlight the use of the ‘`-B`’ option on both boards. This debug option prints the number of buffer overruns during the process, which can occur due to USB saturation or slow memory when recording at high sample rates. While losing a few samples may not significantly impact other applications, in GNSS, it can cause a loss of lock in the tracking loops. Even more critically, dropped samples happen independently on each board, disrupting their synchronization. Therefore, monitoring buffer overruns is essential to ensure successful synchronous recordings.

The synchronization procedure is summarized next:

1) Connect clock-out from the master board to clock-in from the slave board.[1]
2) Enable clock-out sharing in the master board.
3) Connect the trigger output from the master board to the trigger input on the slave board.
4) Arm the trigger in the slave board with the -H option in the normal transfer command.
5) Fire the trigger in the master board by starting the normal transfer command.

[1]For HackRF-based setup, the clock from the master board is used; however, as done for the bladeRF-based setup, an external clock reference could also be used to provide the clock signals for all the boards.

D. Synchronization results

According to the documentation of both SDRs, synchronization is achieved with a maximum difference of 1 sample between boards. To validate this, we tested the procedure using recordings of a step signal and the GNSS spectrum. Both boards were connected to a signal splitter using cables of similar length and configured to the same frequency, sample rate, and gain. A signal was then injected into the splitter source, and synchronous recording was initiated.

Examining the step signal in the time domain (Figure 4), the signal rises at approximately the same time (around sample 60), confirming that the synchronization is working as expected. This behavior is observed consistently with both the bladeRF and HackRF.

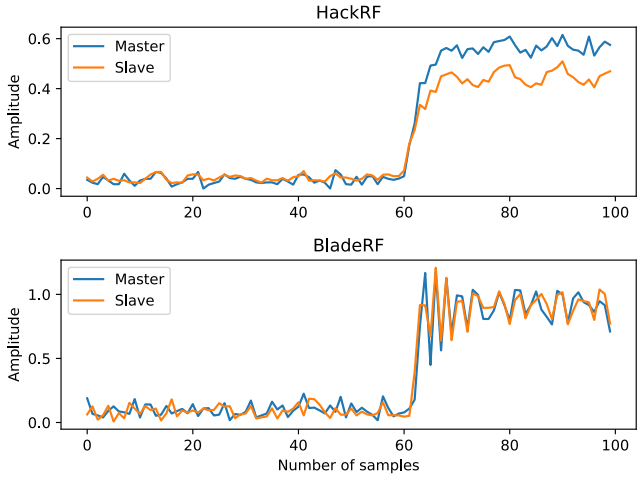

Fig. 4. Recording of a step signal on synchronized bladeRF and HackRF boards.

However, to further validate the synchronization, we recorded the GNSS spectrum at the E1 frequency at 10 Msps. At this frequency, the dominant factor is noise, with the GNSS signal PRN codes buried within it. Correlating both recordings produced a clear tone at 0 offset, confirming that the delay between boards is less than one sample (Figure 5).

III. Application to Galileo SAS

Our goal is to check the consistency between the code delay estimates of the E1-B and E6-C signals. To achieve this, we use a custom-built SAS simulator implemented in MATLAB. The simulator first divides the recorded snapshots into smaller chunks of 4 ms to be processed individually and, for each chunk, performs the acquisition of both E1-B and E6-C using a coherent integration time of 4 ms. Finally, it computes the difference in samples between both code delay estimates to obtain the offset between both bands. A total of 5 seconds per snapshot is used, resulting in the processing of 1250 chunks per snapshot. To achieve sub-sample delay resolution, the simulator implements a peak interpolation during acquisition, as is typical for snapshot receivers [12].

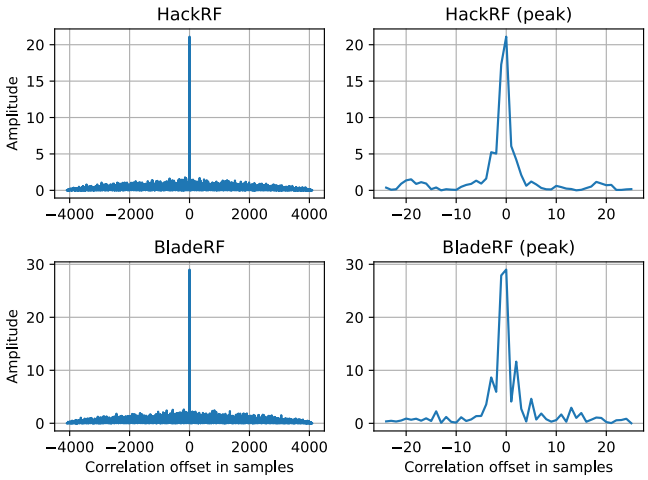

Fig. 5. Correlation of the recording of the GNSS E1 spectrum on synchronized bladeRF and HackRF boards.

In Figure 6 we show the offsets obtained for both setups. A sampling rate of 10 Msps was used to avoid overruns in the HackRF boards. The same configuration was used for the bladeRF boards, including an 8-bit resolution for the IQ samples. As we can observe, the code delay offsets remain within 1 sample.

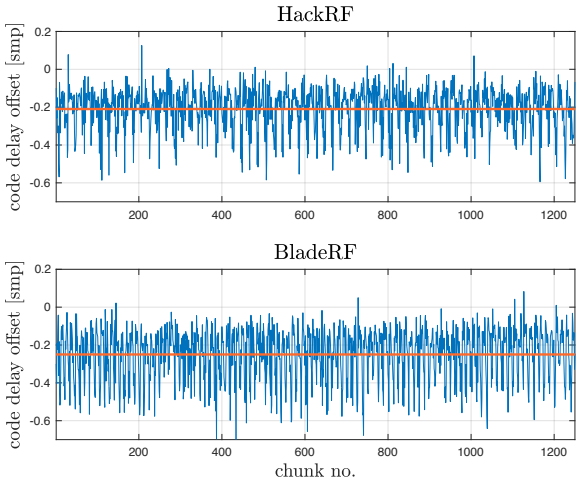

Fig. 6. Code delay offset along the chunks processed.

In Figure 7 and Figure 8, we show the histograms of the estimated offsets for the HackRF and bladeRF setups, respectively. The experimental data is also fitted to a Gaussian-like distribution to estimate the mean and standard deviation for both platforms. The estimated standard deviation for the offsets is approximately 0.1 samples.

It is worth noting that for SAS, a sampling rate of at least 20 Msps is recommended for processing E6-C, as its main lobe spans approximately 10 MHz. The results presented in [4] are based on this configuration on bladeRF.

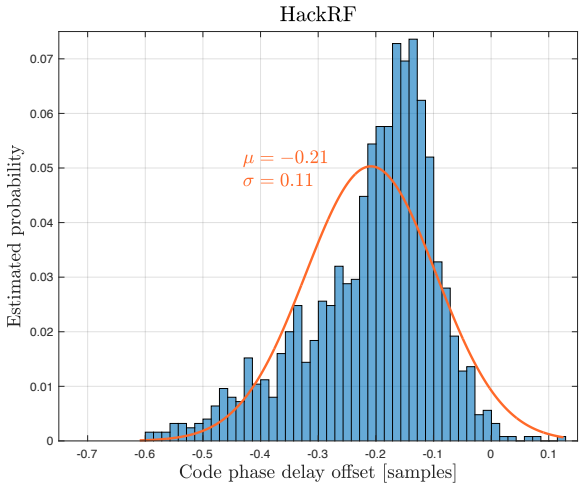

Fig. 7. E6-C vs E1-B code delay offset with HackRF at 10 Msps.

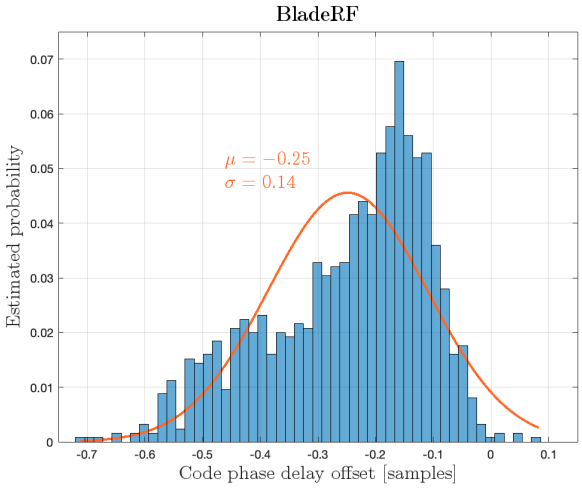

Fig. 8. E6-C vs E1-B code delay offset with bladeRF at 10 Msps.

IV. Conclusion

The experimental results indicate that the synchronization offset between multiple SDRs is limited to less than one sampling period, which is adequate for a wide range of applications, including Galileo SAS. While both platforms tested yielded similar results at the tested sampling rate (10 Msps), the bladeRF-based solution demonstrates superior performance due to its advanced hardware. This enables higher sampling rates with greater accuracy, avoiding overruns.

Although our tests synchronized only two boards, the procedure can be extended to three or more, covering multiple bands. Additional boards should be configured as slaves, receiving the trigger and clock signals from a single master. Alternatively, all boards could be configured as slaves sharing an external clock and a trigger signal (e.g., a rising edge) could be used to initiate operations. Multi-frequency recordings in GNSS mitigate ionospheric errors, improve positioning accuracy, and enhance resilience against interference and jamming.

References

[1] I. Fernandez-Hernandez, J. Winkel, C. O'Driscoll, G. Caparra, R. Terris-Gallego, *et al.*, "Galileo Signal Authentication Service (SAS)," in *37th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2024)*, Baltimore, Maryland, Sep. 2024, pp. 3292 –3307.

[2] 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. 4393-4404, Aug. 2023.

[3] 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 *European Navigation Conference, ESA ESTEC*, Noordwijk, The Netherlands: MDPI, 2023.

[4] 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 *36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*, Denver, Sep. 2023.

[5] M. Bartolucci, J. A. del Peral-Rosado, R. Estatuet-Castillo, J. A. García-Molina, M. Crisci, *et al.*, "Synchronisation of Low-Cost Open Source SDRs for Navigation Applications," in *8th ESA Workshop on Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing (NAVITEC)*, 2016, pp. 1–7.

[6] *Nuand Website – 2023-02 Release*. [Online]. Available: https://www.nuand.com/2023-02-release-122-88mhz-bandwidth.

[7] *Nuand Website – bladeRF 2.0 Micro*. [Online]. Available: https://www.nuand.com/bladerf-2-0-micro/.

[8] *HackRF Website – Technical Documentation*. [Online]. Available: https://hackrf.readthedocs.io/en/latest/index.html.

[9] C. O'Driscoll and J. T. Curran, "Carrier Phase Tracking Considerations for Commodity SDR Hardware," Jul. 2018. DOI: 10.33012/2018.16117.

[10] *Nuand Forum – Synchronized TRX*. [Online]. Available: https://github.com/Nuand/bladeRF/tree/master/host/examples/bladeRF-cli/sync_trx.

[11] *HackRF One – Hardware Triggering*. [Online]. Available: https://hackrf.readthedocs.io/en/latest/hardware_triggering.html.

[12] R. Terris-Gallego, J. A. López-Salcedo, G. Seco-Granados, and I. Fernandez-Hernandez, "Efficient Detection of Galileo ACAS Sequences using E6-B Aiding," in *European Navigation Conference, ESA ESTEC*, Noordwijk, The Netherlands: MDPI, 2024.