# GESTALT: A Testbed For Experimentation And Validation Of GNSS Software Receivers

Javier Arribas, Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), Spain.
Carles Fernández–Prades, Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), Spain.
Pau Closas, Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), Spain.

## BIOGRAPHY

**Dr. Javier Arribas** holds a position of Senior Researcher at the Centre Tecnològic de Telecomunicacions de Catalunya (CTTC). He received the BSc and MSc degree in Telecommunication Engineering in 2002 and 2004, respectively, at La Salle University in Barcelona, Spain. In 2012 he received the PhD degree in Electrical Engineering from Universitat Politècnica de Catalunya. His primary areas of interest include statistical signal processing, GNSS synchronization, detection and estimation theory, software defined receivers, FPGA prototyping and the design of RF front-ends.

**Dr. Carles Fernández–Prades** serves as Head of the Communications Systems Division and Senior Researcher at the Statistical Inference for Communications and Positioning Department of the Centre Tecnològic de Telecomunicacions de Catalunya (CTTC). He received the PhD degree in Electrical Engineering from Universitat Politècnica de Catalunya (UPC) in 2006. His primary areas of interest include signal processing, estimation theory, GNSS and the design of RF front-ends.

**Dr. Pau Closas** is Head of the Statistical Inference for Communications and Positioning Department and Senior Researcher at CTTC. He received the MSc and PhD degrees in Electrical Engineering from Universitat Politècnica de Catalunya in 2003 and 2009, respectively. He also holds a MSc degree in Advanced Mathematics and Mathematical Engineering from UPC since 2014. During 2008 he was Research Visitor at the Stony Brook University, New York, USA. His primary areas of interest include statistical and array signal processing, estimation and detection theory, Bayesian filtering, robustness analysis, and game theory, with applications to positioning systems and wireless communications.

## ABSTRACT

This paper presents a new platform for the experimentation with GNSS signals. It includes a set of commercial of-the-shelf hardware and an open source software, constituting a state-of-the-art platform for research and development of next-generation GNSS receivers. The core of the platform is the GNSS-SDR receiver which has been extended to support multi-band and multi-system operations. As a relevant case of use to validate the research facility, we presented a triple band GNSS-SDR customization capable of receiving four GNSS signals in real-time: GPS L1 C/A, GPS L2CM, Galileo E1b, and Galileo E5a. In addition, we provided detailed descriptions of the receiver architecture, identifying the synchronization challenges of the multi-system satellite channels and providing practical and reproducible solutions. The source code developed to produce this paper has been released under the General Public License, and it is freely available on the Internet.

## 1. INTRODUCTION

This paper presents a new platform for the experimentation with Global Navigation Satellite Systems (GNSS) signals, describing in detail a testbed, so named GESTALT® (Gnss SignAL Testbed), that allows for rigorous, transparent, fair and replicable testing of signal processing algorithms and receiver architectures. The testbed includes hardware, software, and networking components, constituting a state-of-the-art facility for research and development of next-generations GNSS receivers.

The swiftly evolving landscape of GNSS signals and systems demands rapid prototyping tools in order to explore receivers full capability, including radically new uses of those signals. That flexibility is hard to find in todays GNSS receiver technology, mainly driven by application-specific integrated circuits (ASIC) and system-on-chip (SoC) implementations with high development costs and very limited degree of reconfigurability, thus hampering experimentation and fair trials of new approaches. Manufacturers are incorporating new features to their commercial receivers at steady pace: most low-cost, mass-market GNSS receivers are already multi-constellation (GPS and GLONASS) but still not multiband. In contrast, professional receivers are mostly dual-band, with some triple-band model (Novatel's OEM628) already available. In all cases, modern GNSS receivers performance heavily relies on assistance data from external systems (e.g., cellular and WiFi networks) in order to shorten the time-to-first-fix or enhance their navigation performance via the application of high-accuracy algorithms such as Real Time Kinematics (RTK) or Precise Point Positioning (PPP).

In spite of the indubitable interest of multi-frequency receivers [1], an experimentation platform at the signal processing level for dual and triple band GNSS receivers was still missing. The presented facility is equipped with broadband, geodetic grade antennas; GNSS signal generators for controlled experiments; state-of-the-art radio-frequency front-ends able to work concurrently in three GNSS frequency bands, with configurable bandwidth, frequency downshifting and filtering; digitation working at sample rates as high as 80 Msps with 8-bit, coherent I/Q samples; high-speed interfaces to a host computer; and an open source GNSS software receiver in charge of signal processing and generation of suitable outputs in standard formats. A non-exhaustive list of suitable applications that could be deployed in such platform is: signal recording and playback, algorithm development and validation, interference monitoring, array processing, a GNSS reference station, antenna/front-end assessment, reflectometry, low-cost high-accuracy solutions, space weather monitoring, ionospheric mapping, customized guidance of unmanned vehicles or GNSS-based cloud services.

A key aspect of the presented testbed, that makes a difference when compared to other reported facilities (for instance, [2]), is its openness. In addition to the fact that it can be fully operated remotely, the core software receiver engine in charge of all the digital signal processing chain is an open source project with a lively community of users and developers [3]. Accordingly, a partial testbed replication can be done on a limited budget with commodity computers and low cost, over-the-counter antennas and radio-frequency front-ends. This allows both for reproducible research and short assessment and validation times, ultimately shortening the gap between ideas for new uses of GNSS signals and user-driven, market-ready products and services.

In order to show the potential and flexibility of the complete system, Section 4 describes the implementation of a triple-band, multi-system GNSS receiver working with GPS L1 C/A, Galileo E1B, GPS L2C and Galileo E5a signals, and Section 5 provides numerical results obtained with real-life signals. Finally, Section 6 concludes the paper. The source code produced in this research was freely released at GNSS-SDR's GitHub repository [4] under the General Public License.

## 2. A GNSS SIGNAL PROCESSING TESTBED

GESTALT® (GNSS SignAL Testbed) is a CTTC facility equipped with broadband, geodetic grade antennas; GNSS signal generators for controlled experiments; state-of-the-art radio-frequency front-ends able to work concurrently in three GNSS frequency bands, with configurable bandwidth, frequency downshifting and filtering; digitation working at sample rates as high as 80 Msps with 8-bit, coherent I/Q samples; high-speed interfaces to a host computer; and an open source GNSS software receiver in charge of signal processing and generation of suitable outputs in standard formats.

Components can be classified as:

- *Signal sources*, including a set of geodetic-grade NavXpereince 3G+C antennas for real-life signals and IFENs NavX-NCS Professional signal generator for controlled experiments. Antennas are located in a platform at the roof of CTTC's building (see 1, along with a waterproof case housing current injectors for active antennas, and quality RF cables that bring the received signals down to the laboratory.
- A rack at the CTTC's Communication Systems Division Laboratory with a patch panel cabinet that allows easy connection to a *set of radio-frequency front ends*, in change of amplification, filtering, downshifting and conversion from analogue to digital of the received signals. Examples of such front-ends are the triple-band Fraunhofer/TeleOrbit's Flexiband receiver [5], NSL's STEREO dual-band front-end, or Ettus Research's USRP X300. Then, the sampled signal stream is fed into a host computer, in charge of executing a software-defined GNSS receiver, and
- The *host computer*: a Dell's PowerEdge R730 Rack Server, equipped with a NVIDIA Tesla K10 graphics processing unit, as a computing machine executing the software receiver and providing adequate interfaces, including network connectivity. Once the raw sampled signals enters the host computer, a software receiver is in charge of all the signal processing chain up to the output products (GNSS observables, navigation messages, position-velocity-time solutions and any of the intermediate computations). The main server's features are summarized in Table 1.

The signal processing is done with an open source GNSS software defined receiver, described in [6, 7], that allows for easy inclusion of all sort of algorithms (signal processing blocks) and a flexible definition of receiver architectures. Such software, so-called GNSS-SDR, implements a skeleton framework around the signal processing blocks to be tested, so that the block behaves as if already part of the larger system (i.e., a full GNSS receiver). Then, the software receiver can work in real time, if using the antennas and front-ends as signal source, or from raw samples stored in a file. In this way, new algorithms can be rapidly developed and validated using both synthetic signals (with controlled parameters) and real-life signals received at the antenna, allowing the measurement of how the block under test impacts high-level key performance indicators such as time to first fix, precision or accuracy of the navigation solution and, even more, testing completely new receiver architectures (e.g., a receiver for applications based on GNSS reflectometry, or a benchmarking tool for different tracking strategies over the same satellite signal) in relevant environments with a dramatic budget cut when compared to other hardware-based solutions.

A non-exhaustive list of suitable applications that could be deployed in such platform is: signal recording and playback, algorithm development and validation, interference monitoring, array processing, a GNSS reference station, antenna/front-end assessment, reflectometry, low-cost high-accuracy solutions, space weather monitoring, ionospheric mapping, customized guidance of unmanned vehicles or GNSS-based cloud services.

In summary, GESTALT® is future-ready facility for the inception, rapid prototyping and validation of new GNSS receiver architectures, algorithms, applications and services.

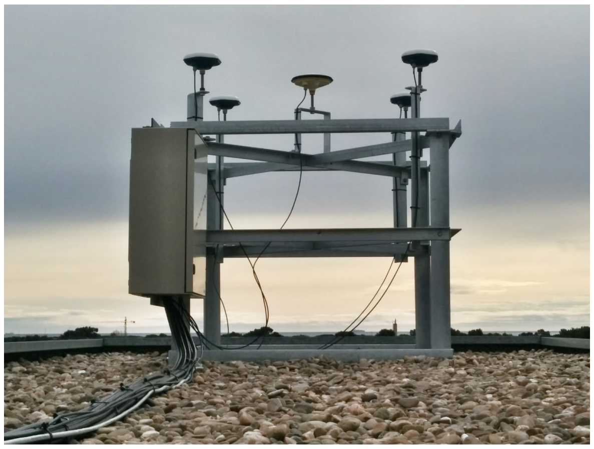


**Fig. 1**: GESTALT®'s rooftop multiband GNSS antenna array facility.

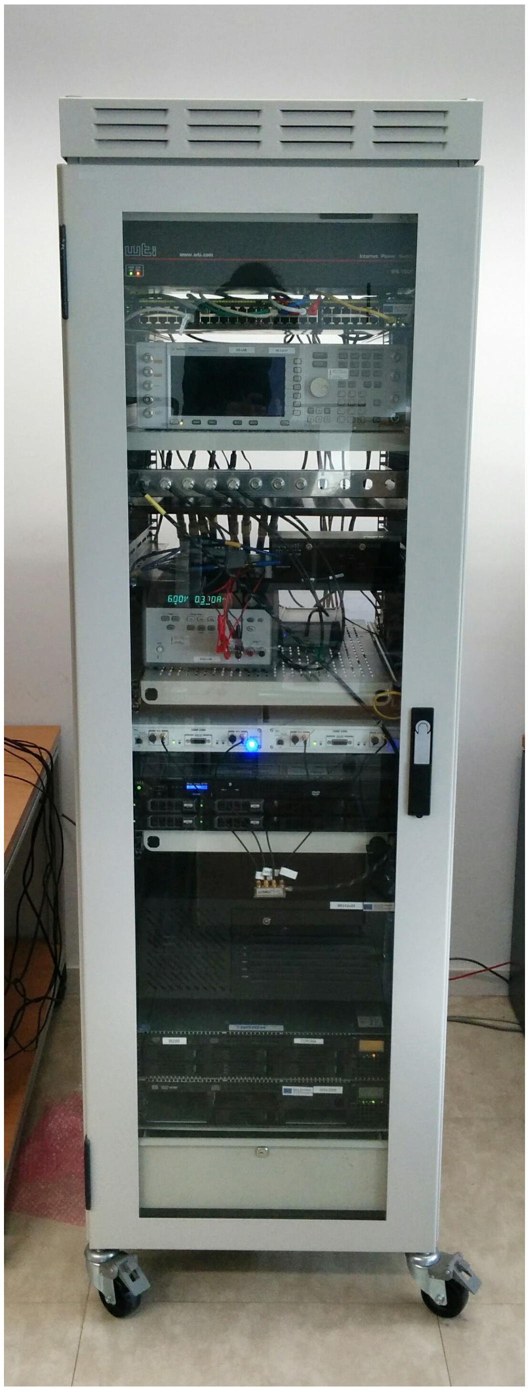


**Fig. 2**: Rack housing the RF front-ends, the signal generator and the computer host.

## 3. SIGNAL MODEL

### 3.1. GPS L1 C/A

Defined in [8], this band is centered at $f_{GPS\,L1} = 1575.42$ MHz. The complex baseband transmitted signal can be writ-

| | |
|---|---|
| Motherboard | 1 PowerEdge R730/xd Motherboard |
| Processors | 2 x Intel Xeon E5-2630 v3 2.4 GHz. |
| | (8 cores, 16 threads each). |
| Memory | 128 GB DDR4 |
| Hard disks | 4x SCSI SAS 1 TB |
| RAID controller | 1 x PERC H730 RAID Controller. |
| | 1 GB cache. |
| Power supply | 1 Dual, Hot-plug, 1100 W |
| GPU | 1 NVIDIA Tesla K10 GPU |
| Network cards | Broadcom 57800 2 x DA/SFP+ 10 Gb. |
| USB Controller | Integrated USB 3.0 controller |

**Table 1**: PowerEdge R730 features.

ten as

$$s_T^{(GPS\,L1)}(t) = e_{L1I}(t) + je_{L1Q}(t)\ , \quad (1)$$

with

$$e_{L1I}(t) = \sum_{l=-\infty}^{\infty} D_{\text{NAV}}\Big[[l]_{204600}\Big] \oplus C_{\text{P(Y)}}\Big[|l|_{L_{\text{P(Y)}}}\Big]p(t - lT_{c,\text{P(Y)}})\ , \quad (2)$$

$$e_{L1Q}(t) = \sum_{l=-\infty}^{\infty} D_{\text{NAV}}\Big[[l]_{20460}\Big] \oplus C_{\text{C/A}}\Big[|l|_{1023}\Big]p(t - lT_{c,\text{C/A}})\ , \quad (3)$$

where $\oplus$ is the exclusive–or operation (modulo–2 addition), $|l|_L$ means $l$ modulo $L$, $[l]_L$ means the integer part of $\frac{l}{L}$, $D_{\text{NAV}}$ is the GPS navigation message bit sequence, transmitted at 50 bps, $T_{c,\text{P(Y)}} = \frac{1}{10.23}$ $\mu$s, $T_{c,\text{C/A}} = \frac{1}{1.023}$ $\mu$s, $L_{\text{P(Y)}} = 6.1871 \cdot 10^{12}$, and $p(t)$ is a rectangular pulse of a chip–period duration centered at $t = 0$ and filtered at the transmitter.

### 3.2. Galileo E1

This band, centered at $f_{Gal\,E1} = 1575.420$ MHz and with a reference bandwidth of 24.5520 MHz, uses the Composite Binary Offset Carrier (CBOC) modulation, defined in baseband as:

$$s_T^{(Gal\,E1)}(t) = \frac{1}{\sqrt{2}}\Big(e_{E1B}(t)\left(\alpha sc_A(t) + \beta sc_B(t)\right) + - e_{E1C}(t)\left(\alpha sc_A(t) - \beta sc_B(t)\right)\Big)\ , \quad (4)$$

where the subcarriers $sc(t)$ are defined as

$$sc_A(t) = \text{sign}\Big(\sin(2\pi f_{s,E1A}t)\Big)\ , \quad (5)$$

$$sc_B(t) = \text{sign}\Big(\sin(2\pi f_{s,E1B}t)\Big)\ , \quad (6)$$

and $f_{s,E1A} = 1.023$ MHz, $f_{s,E1B} = 6.138$ MHz are the subcarrier rates, $\alpha = \sqrt{\frac{10}{11}}$, and $\beta = \sqrt{\frac{1}{11}}$. Channel B contains the I/NAV type of navigation message, $D_{\text{I/NAV}}$, intended for Safety–of–Life (SoL) services:

$$e_{E1B}(t) = \sum_{l=-\infty}^{+\infty} D_{\text{I/NAV}}\Big[[l]_{4092}\Big] \oplus C_{E1B}\Big[|l|_{4092}\Big]p(t - lT_{c,E1B}). \quad (7)$$

In case of channel $C$, it is a pilot (dataless) channel with a secondary code, forming a tiered code:

$$e_{E1C}(t) = \sum_{m=-\infty}^{+\infty} C_{E1Cs}\Big[|m|_{25}\Big] \oplus \sum_{l=1}^{4092} C_{E1Cp}\Big[l\Big] \cdot \cdot\, p(t - mT_{c,E1Cs} - lT_{c,E1Cp})\ , \quad (8)$$

with $T_{c,E1B} = T_{c,E1Cp} = \frac{1}{1.023}$ $\mu$s and $T_{c,E1Cs} = 4$ ms. The $C_{E1B}$ and $C_{E1Cp}$ primary codes are pseudorandom memory code sequences defined in [9, Annex C.7 and C.8]. The binary sequence of the secondary code $C_{E1Cs}$ is 0011100000001010110110010.

### 3.3. GPS L2C

Defined in [8], is only available on Block IIR–M and subsequent satellite blocks. Centered at $f_{GPS\,L2} = 1227.60$ MHz, the signal structure is the same than in (1), with the precision code in the In–phase component, just as in (2) but with an optional presence of the navigation message $D_{\text{NAV}}$. For the Quadrature–phase component, three options are defined:

$$e_{L2CQ}(t) = \sum_{l=-\infty}^{\infty} D_{\text{CNAV}}\Big[[l]_{10230}\Big] \oplus \Big(C_{\text{CL}}\Big[|l|_{L_{\text{CL}}}\Big]p_{1/2}(t - lT_{c,L2C}) + + C_{\text{CM}}\Big[|l|_{L_{\text{CM}}}\Big]p_{1/2}\left(t - \left(l + \frac{3}{4}\right)T_{c,L2C}\right)\Big)\ , \quad (9)$$

$$e_{L2CQ}(t) = \sum_{l=-\infty}^{\infty} D_{\text{NAV}}\Big[[l]_{20460}\Big] \oplus C_{\text{C/A}}\Big[|l|_{1023}\Big]p(t - lT_{c,\text{C/A}})\ , \text{or} \quad (10)$$

$$e_{L2CQ}(t) = \sum_{l=-\infty}^{\infty} C_{\text{C/A}}\Big[|l|_{1023}\Big]p(t - lT_{c,\text{C/A}})\ , \quad (11)$$

where $T_{c,L2C} = \frac{1}{511.5}$ ms and $p_{1/2}(t)$ is a rectangular pulse of half chip–period duration, thus time–multiplexing both codes. The civilian long code $C_{\text{CL}}$ is $L_{\text{CL}} = 767250$ chips long, repeating every 1.5 s, while the civilian moderate code $C_{\text{CM}}$ is $L_{\text{CL}} = 10230$ chips long and its repeats every 20 ms. The CNAV data is an upgraded version of the original NAV navigation message, containing higher precision representation and nominally more accurate data than the NAV data. It is transmitted at 25 bps with forward error correction (FEC) encoding, resulting in 50 sps.

## 3.4. Galileo E5

Centered at $f_{\textit{Gal E5}} = 1191.795$ MHz and with a total bandwidth of $51.150$ MHz, its signal structure deserves some analysis. The AltBOC modulation can be generically expressed as

$$s^{\textit{AltBOC}}(t) = x_1(t)v^*(t) + x_2(t)v(t) \ , \tag{12}$$

where $v(t) = \frac{1}{\sqrt{2}}\left(\text{sign}\left(\cos(2\pi f_s t)\right) + j\text{sign}\left(\sin(2\pi f_s t)\right)\right)$ is the single side–band subcarrier, $f_s$ is the subcarrier frequency, $(\cdot)^*$ stands for the conjugate operation, and $x_1(t)$ and $x_2(t)$ are QPSK signals. The resulting waveform does not exhibit constant envelope. In case of Galileo, the need for high efficiency of the satellites' onboard High Power Amplifier (HPA) has pushed a modification on the signal in order to make it envelope–constant and thus use the HPA at saturation. This can be done by adding some inter–modulation products to the expression in (12), coming up with the following definition:

$$\begin{aligned} s_T^{\textit{(Gal E5)}}(t) =& e_{E5a}(t)ssc_s^*(t) + e_{E5b}(t)ssc_s(t) + \\ &+ \bar{e}_{E5a}(t)ssc_p^*(t) + \bar{e}_{E5b}(t)ssc_p(t) \ , \end{aligned} \tag{13}$$

where the single and product side–band signal subcarriers are

$$ssc_s(t) = sc_s(t) + jsc_s\left(t - \frac{T_s}{4}\right) \ , \tag{14}$$

$$ssc_p(t) = sc_p(t) + jsc_p\left(t - \frac{T_s}{4}\right) \ , \tag{15}$$

and

$$e_{E5a}(t) = e_{E5aI}(t) + je_{E5aQ}(t), \tag{16}$$

$$e_{E5b}(t) = e_{E5bI}(t) + je_{E5bQ}(t), \tag{17}$$

$$\bar{e}_{E5a}(t) = \bar{e}_{E5aI}(t) + j\bar{e}_{E5aQ}(t), \tag{18}$$

$$\bar{e}_{E5b}(t) = \bar{e}_{E5bI}(t) + j\bar{e}_{E5bQ}(t), \tag{19}$$

$$\bar{e}_{E5aI}(t) = e_{E5aQ}(t)e_{E5bI}(t)e_{E5bQ}(t), \tag{20}$$

$$\bar{e}_{E5aQ}(t) = e_{E5aI}(t)e_{E5bI}(t)e_{E5bQ}(t), \tag{21}$$

$$\bar{e}_{E5bI}(t) = e_{E5bQ}(t)e_{E5aI}(t)e_{E5aQ}(t), \tag{22}$$

$$\bar{e}_{E5bQ}(t) = e_{E5bI}(t)e_{E5aI}(t)e_{E5aQ}(t). \tag{23}$$

The signal components are defined as

$$\begin{aligned} e_{E5aI}(t) =& \sum_{m=-\infty}^{+\infty} C_{E5aIs}\Big[|m|_{20}\Big] \oplus \sum_{l=1}^{10230} C_{E5aIp}\Big[l\Big] \oplus \\ &\oplus D_{\text{F/NAV}}\Big[[l]_{204600}\Big]p(t - mT_{c,E5s} - lT_{c,E5p}), \end{aligned} \tag{24}$$

$$\begin{aligned} e_{E5aQ}(t) =& \sum_{m=-\infty}^{+\infty} C_{E5aQs}\Big[|m|_{100}\Big] \oplus \sum_{l=1}^{10230} C_{E5aQp}\Big[l\Big] \cdot \\ &\cdot p(t - mT_{c,E5s} - lT_{c,E5p}), \end{aligned} \tag{25}$$

$$\begin{aligned} e_{E5bI}(t) =& \sum_{m=-\infty}^{+\infty} C_{E5bIs}\Big[|m|_{4}\Big] \oplus \sum_{l=1}^{10230} C_{E5aIp}\Big[l\Big] \oplus \\ &\oplus D_{\text{I/NAV}}\Big[[l]_{40920}\Big]p(t - mT_{c,E5s} - lT_{c,E5p}), \end{aligned} \tag{26}$$

$$\begin{aligned} e_{E5bQ}(t) =& \sum_{m=-\infty}^{+\infty} C_{E5bQs}\Big[|m|_{100}\Big] \oplus \sum_{l=1}^{10230} C_{E5bQp}\Big[l\Big] \cdot \\ &\cdot p(t - mT_{c,E5s} - lT_{c,E5p}), \end{aligned} \tag{27}$$

where $T_{c,E5s} = 1$ ms and $T_{c,E5p} = \frac{1}{10.23}$ $\mu$s. Channel A contains the F/NAV type of navigation message, $D_{\text{F/NAV}}$, intended for the Open Service. The I/NAV message structures for the E5bI and E1B signals use the same page layout. Only page sequencing is different, with page swapping between both components in order to allow a fast reception of data by a dual frequency receiver. The single subcarrier $sc_s(t)$ and the product subcarrier $sc_p(t)$ of expressions (14) and (15) are defined as:

$$\begin{aligned} sc_s(t) =& \frac{\sqrt{2}}{4}\text{sign}\left(\cos\left(2\pi f_s t - \frac{\pi}{4}\right)\right) + \\ &+ \frac{1}{2}\text{sign}\Big(\cos\left(2\pi f_s t\right)\Big) + \\ &+ \frac{\sqrt{2}}{4}\text{sign}\left(\cos\left(2\pi f_s t + \frac{\pi}{4}\right)\right) \ , \end{aligned} \tag{28}$$

$$\begin{aligned} sc_p(t) =& -\frac{\sqrt{2}}{4}\text{sign}\left(\cos\left(2\pi f_s t - \frac{\pi}{4}\right)\right) + \\ &+ \frac{1}{2}\text{sign}\Big(\cos\left(2\pi f_s t\right)\Big) + \\ &- \frac{\sqrt{2}}{4}\text{sign}\left(\cos\left(2\pi f_s t + \frac{\pi}{4}\right)\right) \ , \end{aligned} \tag{29}$$

with a subcarrier frequency of $f_s = 15.345$ MHz. Plotting the power spectrum of the carriers in (13) (see Figure 3), we can see that the QPSK signal $e_{E5a}(t)$ defined in (16) is shifted to $f_{\textit{Gal E5a}} \doteq f_{\textit{Gal E5}} - f_s = 1176.450$ MHz, while $e_{E5b}(t)$ is shifted to $f_{\textit{Gal E5b}} \doteq f_{\textit{Gal E5}} + f_s = 1207.140$ MHz. Thus, we can bandpass filter around $f_{\textit{Gal E5a}}$ and get a good approximation of a QPSK signal, with very low energy components of

$e_{E5b}(t)$, $\bar{e}_{E5a}(t)$, and $\bar{e}_{E5b}(t)$:

$$s_T^{(Gal\ E5a)}(t) \simeq e_{E5aI}(t) + je_{E5aQ}(t). \qquad (30)$$

The same applies to $e_{E5b}(t)$, allowing an independent reception of two QPSK signals and thus requiring considerably less bandwidth than the processing of the whole E5 band.

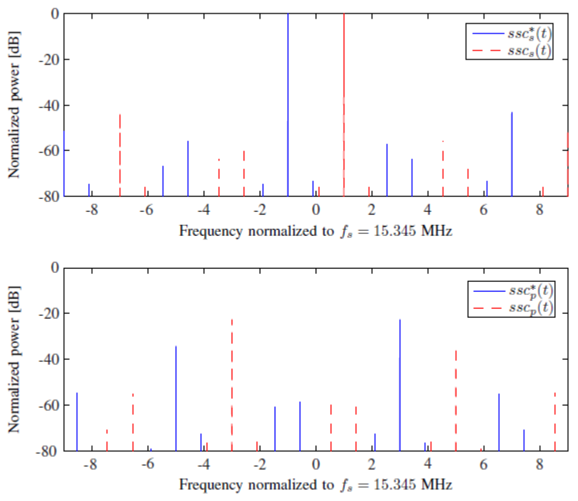


**Fig. 3**: Power spectrum of single and product side-band subcarriers signals in Equation (13), normalized to the power of $ssc_s^*(t)$ at $f_{Gal\ E5a}$.

## 4. GNSS-SDR MULTIBAND RECEIVER ARCHITECTURE

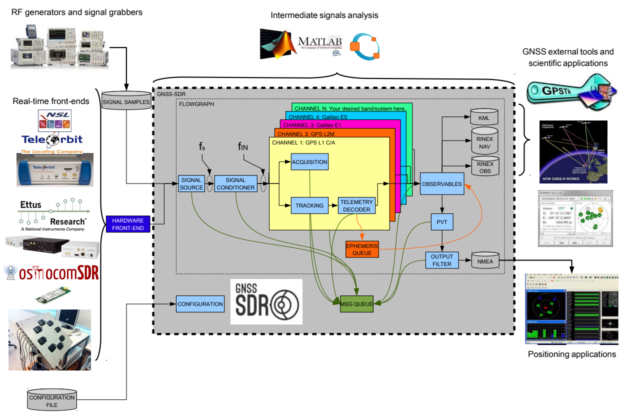


**Fig. 4**: GNSS-SDR block diagram configured for multi-system and multi-band operation.

The general diagram the GNSS-SDR software receiver and its interfaces is shown in Figure 4. From left to right, *Signal Source* can be configured to use a file containing raw data or a radio-frequency front-end. When instantiated, such implementation reads signal samples from the file stored in a disk or from the corresponding bus of the host machine (USB or Ethernet) when a RF front-end is employed, and handles the data stream delivering to the following block in the flow graph. Then, the *Signal Conditioner* is in charge of adapting the data type and performing a FIR filtering and frequency translation that shifts the remnant intermediate frequency to zero Hz. The signal sample stream is then fed to different parallel *Channels* that perform signal detection and synchronization, and the demodulation/decoding of the navigation message. The flexibility of the SDR implementation allows the customization of each of the instantiated channels, thus allowing the setup of a custom multisystem / multiband receiver without modifying the source code but using the configuration file.

When GNSS-SDR is configured as a multiband/multi-channel receiver, there are two possibilities regarding the front-end hardware:

- A front-end device equipped with multiple channels or bands
- Multiple front-end devices synchronized externally

### 4.1. Multichannel front-end

A software Signal Source block can be equipped with more than one radio-frequency chain. Examples of such configuration could be a USRP [10] with two subdevices, or dual or triple band RF front ends, such as NSL Stereo or a TeleOrbit Flexiband [5]. This case implies not only the configuration of the Signal Source, but also there is a need to set up different Signal Conditioners for each band, and configure the Channel implementations for the different signals present on each band, as shown in Fig 5.

Listing 1: Configuration entry for a dual-band receiver.

```
SignalSource.RF_channels=2
```

Then:

Listing 2: Configuring a dual-band GPS L1/L2 receiver.

```
SignalSource.RF_channels=2
SignalSource.implementation=UHD_Signal_Source
...
SignalSource.subdevice=A:0 B:0
...
SignalSource.freq0=1575420000
SignalSource.freq1=1227600000
...
SignalConditioner0.implementation=...
DataTypeAdapter0.implementation=...
InputFilter0.implementation=...
Resampler0.implementation=...
```

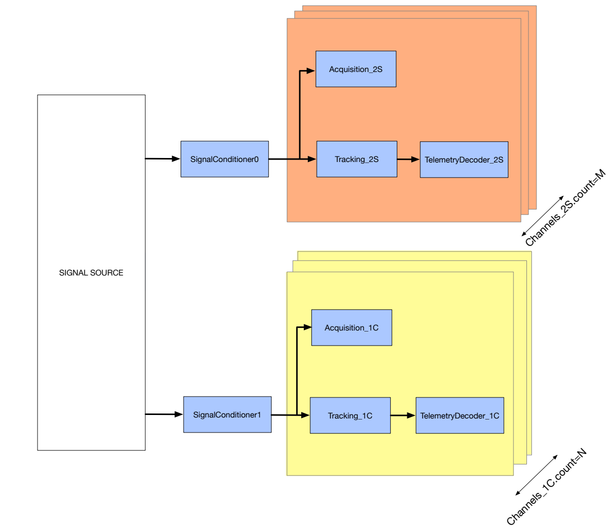


**Fig. 5**: Simplified block diagram of a dual-band receiver of GPS L1 C/A and GPS L2C (M) signals.

```
SignalConditioner1.implementation=...
DataTypeAdapter1.implementation=...
InputFilter1.implementation=...
Resampler1.implementation=...
...
Channels_1C.count=8
Channels_2S.count=8

; # Channel connection
Channel0.RF_channel_ID=1
Channel1.RF_channel_ID=1
Channel2.RF_channel_ID=1
Channel3.RF_channel_ID=1
Channel8.RF_channel_ID=0
Channel9.RF_channel_ID=0
Channel10.RF_channel_ID=0
Channel11.RF_channel_ID=0

; Channel signal
Channel0.signal=1C
Channel1.signal=1C
Channel2.signal=1C
Channel3.signal=1C
Channel4.signal=2S
Channel5.signal=2S
Channel6.signal=2S
Channel7.signal=2S
...
Acquisition_1C.implementation=...
    ; or Acquisition_1C0, ..., Acquisition_1C3
Acquisition_2S.implementation=...
    ; or Acquisition_2S4, ..., Acquisition_2S8


Tracking_1C.implementation=...
    ; or Tracking_1C0, ..., Tracking_1C3
Tracking_2S.implementation=...
    ; or Tracking_2S4, ..., Tracking_2S8


TelemetryDecoder_1C.implementation=...
    ; or TelemetryDecoder_1C0, ...,
    TelemetryDecoder_1C3
TelemetryDecoder_2S.implementation=...
    ; or TelemetryDecoder_2S4, ...,
    TelemetryDecoder_2S8
...
```

### 4.2. Multiple front-end devices

On the other hand, GNSS-SDR can be configured with more than one signal source delivering signal sample streams simultaneously. Notice that in this mode of operation, the front-end devices require an external synchronization mechanism for both the sampling clock and the their local oscillators. Examples of multiple front-end sources configuration could be:

- Two files, one for each band (such as in the case of NSL's Stereo front-end);
- Different antennas, working at the same band but with different RF front-ends;
- Different front-ends sharing the same antenna.

In order to complete the signal chains, each of the front-end signal source block need to have its own signal conditioner instantiated, as shown in Fig. 6

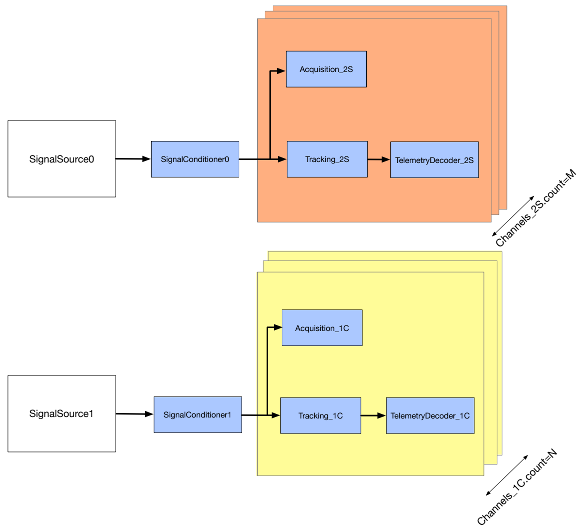


**Fig. 6**: Simplified block diagram of a multi-source receiver of GPS L1 C/A and GPS L2C (M) signals.

The number of radio-frequency chains is denoted by parameter `RF_channels`, which defaults to one if it is not present in the configuration file.

Listing 3: Configuration entry for two signal sources.

```
Receiver.sources_count=2
```

Then:

Listing 4: Configuring a multiple source receiver.

```
Receiver.sources_count=2
...
SignalSource0.implementation=...
SignalSource1.implementation=...
...
SignalConditioner0.implementation=...
DataTypeAdaper0.implementation=...
InputFilter0.implementation=...
...
SignalConditioner1.implementation=...
DataTypeAdaper1.implementation=...
InputFilter1.implementation=...
...
Channels_1C.count=2
Channels_1B.count=2
...
; # CHANNEL CONNECTION
Channel0.SignalSource_ID=0
Channel1.SignalSource_ID=0
Channel2.SignalSource_ID=1
Channel3.SignalSource_ID=1

Channel0.signal=1C
Channel1.signal=1C
Channel2.signal=1B
Channel3.signal=1B
...
```

## 4.3. CHANNEL SYNCHRONIZATION IN A MULTI-BAND/MULTISYSTEM SDR RECEIVER

A mandatory requirement for any set of front-ends simultaneously connected to the receiver is the phase coherence between RF channels. This constraint prevents the introduction of random bias in the computed GNSS observables. It implies a phase synchronization of all the downconverters local oscillators and a phase synchronization of the ADCs sample clocks. A technological solution for this problem is to distribute a common clock reference signal. In addition, in the scenario of multiple front-end devices, it is required to have a time-tag attached to each sample stream in order to take into account the external hardware buffer delays.

Fortunately, almost all the USRP networked series of frontends and the TeleOrbit Flexiband front-end had been engineered with the these constraints in mind, thus, it is possible to activate the synchronization features directly from the device driver [11].

However, even assuming a perfect synchronization between all the signal paths, there is a new synchronization problem arisen inside the SDR receiver: the update of code delay, carrier phase estimations, and Doppler frequencies estimated by DLLs and PLLs, for each of the tracked GNSS signals are not synchronized in time between channels.

Examining the GNSS signals structures shown in Section 3, the main causes of this problem are:

- Differences in the duration of the spreading code sequences.
- Different coherent integration times for the tracking correlators.
- Different reception times between channels, due to the delays caused by the internal buffers and the multi-thread channel processing, as reported in [7].

Regarding the spreading code sequences, Table 2 lists the primary spreading sequence length and its associated duration for GPS and Galileo civil signals. Notice that the shortest coherent integration time for a complete primary sequence is 1 ms, shared by GPS L1 C/A and Galileo E5. Assuming a DLL/PLL signal tracking using only the primary code, it would produce estimations at a rate of 1 kHz. On the other hand, a GPS L2CM channel will produce estimations every 20 ms, that is a rate of 50 Hz.

A feasible solution is the resampling of the channel estimations to synchronize inputs to the observable computation algorithm (i.e. code pseudorranges and carrier phases) implemented in the observables block (see Fig. 4). It is possible to enable this feature in the telemetry decoder block by specifying a decimation rate for each channel in the receiver configuration file as follows:

Listing 5: Configuring the channel output decimation rate in a triple band GPS L1 C/A + Galileo E1 + GPS L2CM + Galileo E5 receiver. Notice that a decimation factor of 20 in the GPS L1 C/A telemetry decoder output and decimation of 5 in Galileo E1B will equalize the outputs to the observables block at a rate of 20 ms (50 Hz).

```
TelemetryDecoder_1C.implementation=...
GPS_L1_CA_Telemetry_Decoder
TelemetryDecoder_1C.decimation_factor=20;

TelemetryDecoder_1B.implementation=...
Galileo_E1B_Telemetry_Decoder
TelemetryDecoder_1B.decimation_factor=5;

TelemetryDecoder_2S.implementation=...
GPS_L2_M_Telemetry_Decoder
TelemetryDecoder_2S.decimation_factor=1;

TelemetryDecoder_5X.implementation=...
Galileo_E5a_Telemetry_Decoder
TelemetryDecoder_5X.decimation_factor=1;
...
```

## 4.4. MIXED GNSS OBSERVABLES COMPUTATION

GNSS-SDR generates code observables (pseudoranges) based on setting a common reception time across all channels [12]. The result of this approach is not an absolute pseudorange, but a *relative* pseudorange with respect to the value (of

| Signal | PRN length rate [chips] | Primary code [MHz] | Duration [s] |
|---|---|---|---|
| **GPS L1 C/A** | 1023 | 1.023 | 0.001 |
| **GPS L2CM** | 10230 | 0.5115 | 0.02 |
| **GPS L2CL** | 767250 | 0.5115 | 1.5 |
| **Galileo E1** | 4092 | 1.023 | 0.004 |
| **Galileo E5** | 10230 | 10.230 | 0.001 |

**Table 2**: Primary spreading sequence length, chip rate, and duration for the primary codes of GPS and Galileo civil signals.

pseudorange) allocated for a reference satellite. This is possible thanks to the TOW information, that is the epoch denoted in the navigation message, and the associated reception time $t_{RX}$, that is the epoch denoted by the receiver time counter, both available for each satellite.

According to the Receiver Independent Exchange Format (RINEX) v3.02 specification [13], the pseudorange is defined as *the difference of the time of reception (expressed in the time frame of the receiver) and the time of transmission (expressed in the time frame of the satellite) of a distinct satellite signal.* In mixed GNSS observable files, further clarification is introduced: *in a mixed-mode GPS/GLONASS/Galileo/QZSS/BDS receiver referring all pseudorange observations to one receiver clock only*, thus,

- The raw GLONASS pseudoranges will show the current number of leap seconds between GPS/GAL/BDT time and GLONASS time if the receiver clock is running in the GPS, GAL or BDT time frame
- The raw GPS, Galileo and BDS pseudoranges will show the negative number of leap seconds between GPS/GAL/BDT time and GLONASS time if the receiver clock is running in the GLONASS time frame

Leap seconds between GNSS systems can introduce large bias in the pseudoranges and it can overflow the code observations format field. To avoid misunderstandings, RINEX mixed observables requires a correction in the pseudorranges. Since GNSS-SDR uses GPS time as the timeframe for the receiver clock, the GLONASS psudorange should be corrected as follows:

$$\tilde{\rho}_{\text{GLO}} = \rho_{\text{GLO}} - c\Delta tLS \tag{31}$$

where $\rho_{\text{GLO}}$ is the raw GLONASS psudorange, $c$ is the speed of light constant and $\Delta tLS$ stands for the actual number of leap seconds between between GPS/Galileo and GLONASS time, as broadcast in the GPS almanac.

## 5. NUMERICAL RESULTS WITH REAL-LIFE SIGNALS

In this work, we validated the GNSS-SDR architecture for multichannel / multiband receiver configurations. As a relevant case of use, we implemented a triple-band, four signals GNSS receiver capable of acquire and track GPS L1 C/A, Galileo E1B, GPS L2CM, and Galileo E5a in real-time.

The selected RF frontend for this setup was the TeleOrbit Flexiband GTEC frontend. It receives up to three navigation bands in its default hardware configuration: L1/E1, L2 and L5/E5a. The sampled signals are accessible via one USB3.0 interface and an optional parallel port. Table 3 shows the frontend configuration parameters values used in the experiment. The receiver operation parameters stored in the configuration file are listed in Table 4. In order to reduce the required computing power, the original frontend sampling frequency of 20 MSPS was downsampled by a decimating FIR filters, configured for each frontend band. GPS L1 C/A, Galileo E1b, and GPS L2CM bands are downsampled to $f_s = 2.5$ MSPS, while the wider signal Galileo E5a was downsampled to $f_s = 10$ MSPS.

| **RF band** | **L1/E1bc** | **L2/L2C** | **L5/E5a** |
|---|---|---|---|
| **Bandwidth** | 18 MHz | 18 MHz | 18 MHz |
| **Center freq.** | 1575.420 MHz | 1227.600 MHz | 1176.450 MHz |
| **Inter. freq.** | -0.205 MHz | 0.100 MHz | 0.050 MHz |
| **Sampling rate** | 20 Msps | 20 Msps | 20 Msps |
| **Sample bit width** | 2x2bits (complex) | 2x2bits (complex) | 2x4bits (complex) |
| **USB data rate** | 320 MBit/s | | |
| **USB type** | 3.0 | | |

**Table 3**: Flexiband's III-1a triple band configuration.

### 5.1. SIGNAL TRACKING

In order to debug the signal tracking operation, all internal and external outputs from the GNSS-SDR tracking blocks can be recorded in real-time, even in a live signal reception operation by activating in the configuration file the debug dump functionality. Currently it is possible to record the following parameters:

- Correlator's absolute value: $\sqrt{I_{VE}^2 + Q_{VE}^2}$, $\sqrt{I_E^2 + Q_E^2}$, $\sqrt{I_P^2 + Q_P^2}$, $\sqrt{I_L^2 + Q_L^2}$, and $\sqrt{I_{VL}^2 + Q_{VL}^2}$ when available
- $I_P$ In-Phase Prompt correlator value

| **GNSS-SDR configuration** | | | | |
|---|---|---|---|---|
| System | GPS L1 C/A | Galileo E1B | GPS L2CM | Galileo E5A |
| Signal Source | NavXperience 3G+C antenna + TeleOrbit Flexiband front-end (III-1b configuration, see Table 3). Using custom GNURadio-compliant USB 3.0 driver. Implementation: `Flexiband_Signal_Source` Center frequencies: $f_{c1} = 1575.420$ MHz $f_{c2} = 1227.600$ MHz $f_{c3} = 1176.450$ MHz Sampling frequency: $f_s = 20.00$ Msps RF gain: AGC active Sample data type: Internal conversion from 2x2 bits, 2x2 bits, and 2x4 bits to `std::complex<float>` (IEEE-754 32 bit floating point) | | | |
| Signal Conditioner | Data type adapter: No adapter (Implementation: `Pass_Through`) | | | |
| | Input filters: a frequency-translating FIR filter (Parks-McClellan design using Remez algorithm) per frontend band. Implementation: `Freq_Xlating_Fir_Filter` Frequency at the band edges: $\left[0.0\ 0.45\ 0.55\ 1\right] \cdot \frac{f_s}{2}$; amplitude at the band edges: $\left[1\ 1\ 0\ 0\right]$. Number of taps: 5; grid density: 16 Intermediate frequency: $f_{if1} = -0.205$ MHz, $f_{if2} = 0.100$ MHz, $f_{if3} = 0.050$ MHz FIR decimation factor: $d_1 = 8$, $d_2 = 8$, $d_3 = 2$ Sampling rates at FIR output $f_{s1} = 2.5$ MHz, $f_{s2} = 2.5$ MHz, $f_{s3} = 10$ MHz | | | |
| | Resampler: No additional resampling (Implementation: `Pass_Through`) | | | |
| Acquisition | Integration time: 1 ms Implementation: `GPS_L1_CA_PCPS_PCPS_Acquisition` Acq. threshold: 0.008 Doppler range: $\pm 5$ kHz Doppler step: 250 Hz | Integration time: 4 ms Implementation: `Galileo_E1_PCPS_Ambiguous_Acquisition` False alarm probability: $10^{-7}$ Doppler range: $\pm 5$ kHz Doppler step: 125 Hz | Integration time: 20 ms Implementation: `GPS_L2_M_PCPS_PCPS_Acquisition` Acq. threshold: 0.005 Doppler range: $\pm 5$ kHz Doppler step: 30 Hz | Integration time: 1 ms Implementation: `Galileo_E5a_Noncoherent_IQ_Acquisition_CAF` Acq. threshold: 0.002 Doppler range: $\pm 5$ kHz Doppler step: 250 Hz |
| Tracking | Implementation: `GPS_L1_CA_DLL_PLL_Tracking` $BW_{DLL} = 3.0$ Hz $BW_{PLL} = 40$ Hz Loop filters order: 3 Early-Late spacing: 0.5 [chip period] | Implementation: `Galileo_E1_DLL_PLL_VEML_Tracking` $BW_{DLL} = 2$ Hz $BW_{PLL} = 15$ Hz Loop filters order: 3 Early-Late spacing: 0.15 [chip period] V.Early - V.Late spacing: 0.6 [chip period] | Implementation: `GPS_L2_M_DLL_PLL_Tracking` $BW_{DLL} = 0.3$ Hz $BW_{PLL} = 1.5$ Hz Loop filters order: 3 Early-Late spacing: 0.5 [chip period] | Implementation: `Galileo_E5a_DLL_PLL_Tracking` $BW_{DLL} = 2$ Hz $BW_{PLL} = 20$ Hz Loop filters order: 2 Early-Late spacing: 0.5 [chip period] |
| Telemetry Decoder | Implementation: `GPS_L1_CA_Telemetry_Decoder` Decimation factor: 20 | Implementation: `Galileo_E1B_Telemetry_Decoder` Decimation factor: 5 | Implementation: `GPS_L2CM_Telemetry_Decoder` Decimation factor: 1 | Implementation: `Galileo_E5a_Telemetry_Decoder` Decimation factor: 1 |
| Observables | Implementation: `Mixed_Observables` | | | |
| PVT | Least squares solution combining GPS/Galileo pseudoranges through the GGTO. Implementation: `Mixed_PVT` Moving average depth: no average. Output rate: 10 ms | | | |

**Table 4**: Software receiver's configuration parameters used in the experiments with real-life GNSS signals reported in this paper.

- $Q_P$ Quadrature Prompt correlator value
- Sample counter [samples]
- Accumulated carrier phase [Rad]
- Carrier Doppler estimation [Hz]
- Code frequency estimation [Chips/s]
- Raw carrier discriminator output
- Raw code discriminator output
- Filtered carrier discriminator output
- Filtered code discriminator output
- Carrier-to-noise density ratio (CN0) estimation [dB-Hz]
- Carrier lock test statistics
- Remnant code phase [samples]

By using the debug binary log data files, we inspected the acquisition to tracking transition and the time evolution of the DLLs/PLLs transitory operations from signal capture (pull-in) to stable tracking. Figure 8 shows the scatter plot and the Early,Prompt,Late correlator output for a channel tracking a GPS L1 C/A satellite. The DLL/PLL signal capture transitory is short due to the short integration time and the high bandwidth of the PLL filter. On the other hand, we can appreciate a longer transitory effect in a GPS L2CM channel, shown on Figure 9. In this case, the integration time 20 times longer and thus, the DLL/PLL filter bandwidths are reduced in an order of magnitude.

Regarding the Galileo signals, Figure 10 shows the five correlators evolution (Very Early, Prompt, Late, Very Late) for a Galileo E1 satellite (PRN 14 FOC-FM2). A remarkable effect is the overlapping of the Early and Late correlator values with the Prompt ones. This effect is caused by the bare minimum sampling frequency selected for the GPS L1 / Galileo E1 band, which was set to 2.5 MHz. It is enough to keep the track of Galileo E1 signal, but not to discriminate the Early-Late spacing of 0.15 chips in the local signal replica generation. In addition, there is a long tracking pull-in transitory effect, mainly caused by the poor signal bandwidth but also due to the fact that the integration period is 4 ms and thus, the DLL/PLL filters are narrower than the equivalent GPS L1 ones. Finally, Galileo E5a tracking results are shown on Figure 11. With an integration period of 1 ms and enough bandwidth, the records show a very stable tracking with a short pull-in transitory.

To complete the analysis, Figure 7 shows the normalized In-Phase signal of the prompt correlator outputs of three active receiver channels: GPS L1 C/A in blue, GPS L2CM in red and Galileo E1 in green. As shown in Section 4 each channel keeps the track of an internal receiver time, relative to the start of receiver operation. This timestamp is associated to every estimation or product of the tracking block. In the figure, the timestamp is shown in the $X$ axis. The differences in the spreading sequence lengths and duration are clearly visible. For each GPS L2CM correlator output there are approximately 5 Galileo E1 outputs and 20 GPS L1 C/A outputs. Notice also the presence of the BPSK symbols transition of the telemetry message (bits of the navigation message).

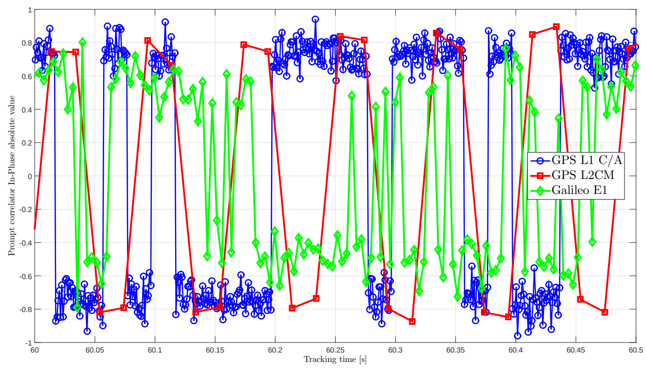
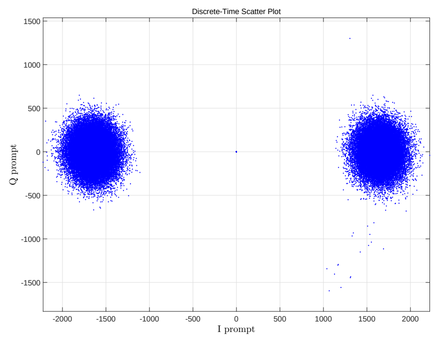


**Fig. 7**: Raw prompt correlator output comparative for a GPS L1 C/A, GPS L2CM and Galileo E1 real-life satellite signals.

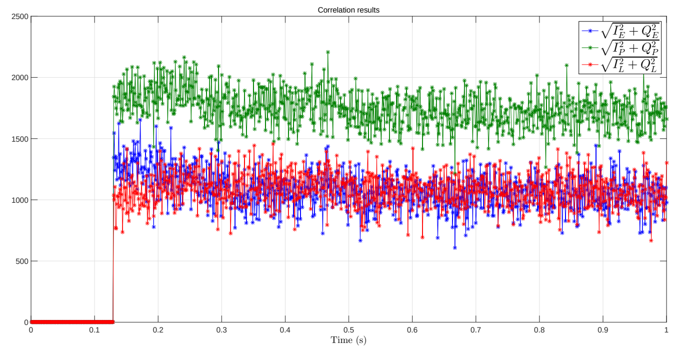
## 6. CONCLUSIONS

This paper presented a new platform, so called GESTALT, for the experimentation with GNSS signals. It includes a set of commercial of-the-shelf hardware and an open source software, constituting a state-of-the-art platform for research and development of next-generation GNSS receivers. The core of the platform is the GNSS-SDR receiver which has been extended to support multi-band and multi-system operations. As a relevant case of use to validate the research facility, we presented a triple band GNSS-SDR customization capable of receiving four GNSS signals in real-time: GPS L1 C/A, GPS L2CM, Galileo E1b, and Galileo E5a. In addition, we provided detailed descriptions of the receiver architecture, identifying the synchronization challenges of the multi-system satellite channels and providing practical solutions. Finally, we included receiver configuration examples to enable the replication of the results presented in this work.

The source code developed to produce this paper has been released under the General Public License (GPL) v3, thus securing practical usability, inspection, and continuous improvement by the research community, allowing the discussion based on tangible code and the analysis of results obtained with real signals. Please check GNSS-SDR's website

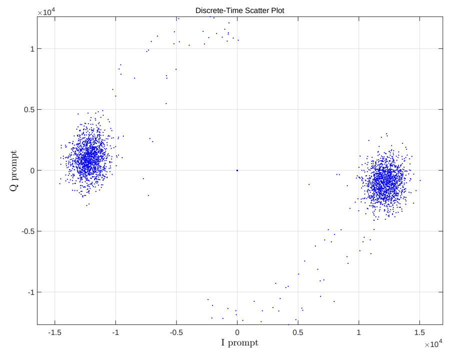


(a) Prompt correlator scatter Plot

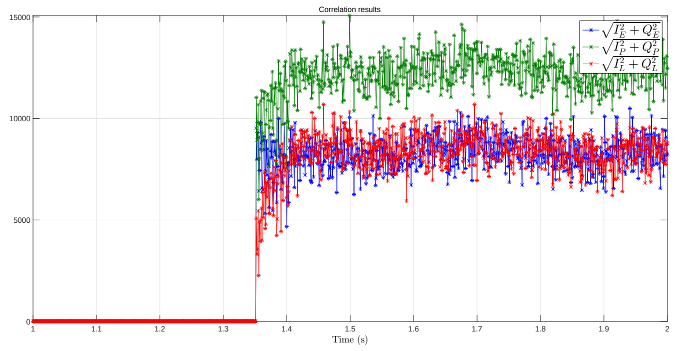


(b) Early, Prompt, and Late correlators output

**Fig. 8**: GPS L1 C/A tracking results

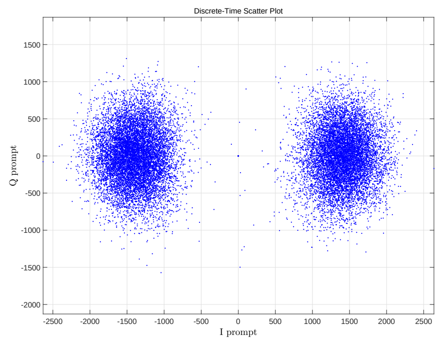


(a) Prompt correlator scatter Plot

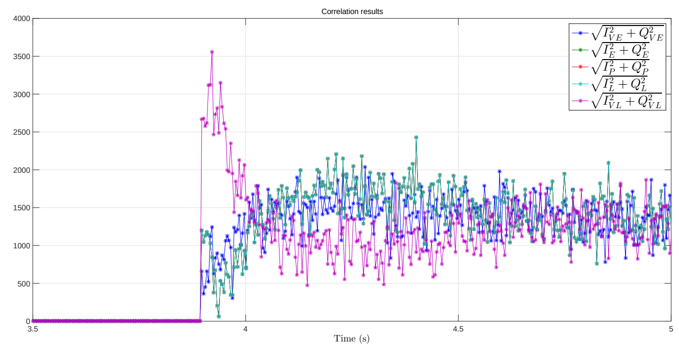


(b) Early, Prompt, and Late correlators output

**Fig. 9**: GPS L2CM tracking results

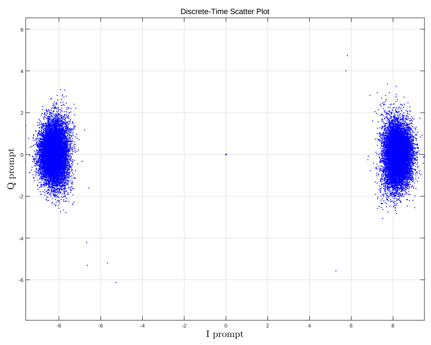


(a) Prompt correlator scatter Plot

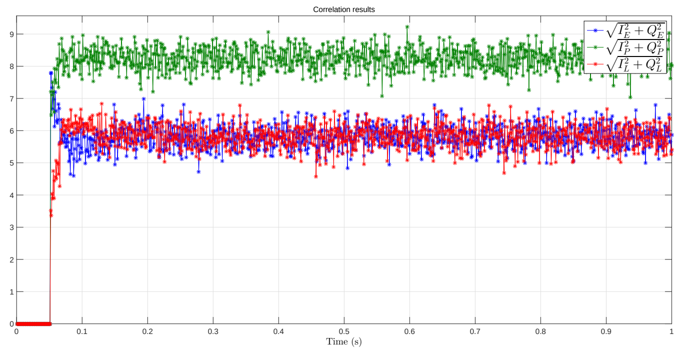


(b) Very Early, Early, Prompt, Late, and Very Late correlators output

**Fig. 10**: Galileo E1b tracking results



(a) Prompt correlator scatter Plot



(b) Early, Prompt, and Late correlators output

**Fig. 11**: Galileo E5a tracking results

at `http://gnss-sdr.org` for more information on how to build the source code and use the software.

## 7. ACKNOWLEDGEMENTS

This work has been possible thanks to many people from the open source community. The authors are as grateful to them as to the institutions and companies that made it possible. GESTALT® has been partially financed by the Operational European Regional Development Fund Programme Catalonia 2007-2013. The work was also supported by the European Commission in the framework of the FP7 Network of Excellence in Wireless COMmunications NEWCOM# (Grant agreement no. 318306), by Google and their Summer of Code Program in 2013, 2014 and 2015, by Nottingham Scientific Limited (NSL), by the Spanish Ministry of Economy and Competitiveness through the project TEC2012-39143 (SOSRAD), and by the Catalan Department of Economy and Knowledge through the recognition of the Research Group 2014-SGR-1567.

## REFERENCES

[1] P. G. Mattos, "Markets and multi-frequency GNSS," *Inside GNSS*, vol. 8, no. 1, pp. 34–37, Jan/Feb. 2013.

[2] B. Huang, Z. Yao, F. Guo, S. Deng, X. Cui, and M. Lu, "STARx A GPU based multi-system full-band real-time GNSS software receiver," in *Proc. of the ION-GNSS+*, Nashville, Tennessee, Sept. 2013.

[3] "GNSS-SDR's Webpage," `http://gnss-sdr.org`, Retrieved: December 24, 2016.

[4] "GNSS-SDR's GitHub Organization Page," `https://github.com/gnss-sdr/gnss-sdr`, Retrieved: December 24, 2016.

[5] A. Rügamer, F. Förster, M. Stahl, and G. Rohmer, "Features and applications of the adaptable Flexiband USB3.0 front-end," in *Proc. of the ION-GNSS+*, Tampa, Florida, Sept. 2014.

[6] C. Fernández–Prades, J. Arribas, P. Closas, C. Avilés, and L. Esteve, "GNSS-SDR: An open source tool for researchers and developers," in *Proc. of the ION GNSS 2011 Conference*, Portland, Oregon, Sept. 2011.

[7] J. Arribas, M. Branzanti, C. Fernández-Prades, and P. Closas, "Fastening GPS and Galileo tight with a software receiver," in *Proc. of the ION-GNSS+*, Tampa, Florida, Sept. 2014.

[8] Global Positioning Systems Directorate, *Interface Specification IS-GPS-200 Revision H. Navstar GPS Space Segment/Navigation User Interfaces*, Sep. 24, 2013.

[9] European Union, *European GNSS (Galileo) Open Service. Signal In Space Interface Control Document. Ref: OS SIS ICD, Issue 1.1*, September 2010.

[10] Ettus Research, "USRP family of products," `https://www.ettus.com/product`, Retrieved: December 24, 2016.

[11] Ettus Research, "UHD - USRP Hardware Driver," `http://files.ettus.com/uhd_docs/manual/html/`, Retrieved: December 24, 2016.

[12] M. Petovello M. Rao, G. Falca, "Code Tracking & Pseudoranges," *Inside GNSS*, vol. 7, pp. 26–33, January 2012.

[13] International GNSS Service (IGS), RINEX Working Group and Radio Technical Commission for Maritime Services Special Committee 104 (RTCM-SC104), *The Receiver Independent Exchange Format (RINEX) v 3.02*, April 2013.