**IAES International Journal of Artificial Intelligence (IJ-AI)**
Vol. 13, No. 4, December 2024, pp. 3869~3878
ISSN: 2252-8938, DOI: 10.11591/ijai.v13.i4.pp3869-3878

# Implementation of global navigation satellite system software-defined radio baseband processing algorithms in system on chip

**Chetna Devi Kh[1], Malode Vishwanatha Panduranga Rao[2]**
[1]Department of Electronics Engineering, Faculty of Engineering and Technology, JAIN (Deemed-to-be University), Bengaluru, India
[2]Department of Computer Science and Engineering, Faculty of Engineering and Technology, JAIN (Deemed-to-be University), Bengaluru, India

**Article Info**

***Article history:***

Received Feb 12, 2024
Revised May 23, 2024
Accepted Jun 1, 2024

***Keywords:***

Baseband processing
Global navigation satellite system receiver
Software defined radio
USRP N210
Zynq-system on chip

**ABSTRACT**

The global navigation satellite system (GNSS) is an international navigation system that determines users' locations globally using a constellation of satellites. Conventional hardware-based receivers often face challenges related to cost-effectiveness and lack of reconfigurability. To address these issues, GNSS software receivers have emerged, executing baseband processing methods on host computers. However, host PC-based GNSS software receivers encounter obstacles during real-time signal acquisition, such as computational complexity and data loss. This research paper introduces a real-time system on chip (SoC)-based GNSS software receiver to mitigate these concerns. The receiver utilizes the USRP N210 radio frequency (RF) front end to acquire GNSS signals in real-time. Baseband processing algorithms are executed using the Zynq 7000 SoC board, with modifications applied to the acquisition module. The effectiveness of the SoC-based GNSS receiver is evaluated under both stationary and dynamic conditions. Experimental outcomes indicate that the receiver provides precise user localization and facilitates prototype development. This methodology not only overcomes the limitations of conventional hardware-based receivers but also leverages the benefits of SoC architecture to process GNSS signals in a flexible and efficient manner.

***Corresponding Author:***

Chetna Devi Kh
Department of Electronics Engineering, Faculty of Engineering and Technology
JAIN (Deemed-to-be University)
45th km, NH - 209, Jakkasandra Post, Bengaluru - Kanakapura Rd, Bengaluru, Karnataka 562112, India
Email: choisiwon.kh@gmail.com

## 1. INTRODUCTION

The incorporation of global navigation satellite system (GNSS) software receivers is crucial in contemporary navigation systems, as they provide a flexible and programmable method for obtaining accurate time, location, and navigation data [1], [2]. In contrast to their conventional hardware-based counterparts, GNSS software receivers function using computational algorithms to analyse signals obtained from a constellation of satellites [3]. These receivers operate on software-defined principles. The advent of software-defined receivers has sparked a significant paradigm shift in the field of satellite navigation, facilitating the development of flexible and individualised approaches to signal processing. By utilising advanced algorithms and digital signal processing techniques, these software receivers effectively decode, demodulate, and derive vital navigation data from signals that are transmitted by GNSS satellites [4].

By virtue of its intrinsic adaptability, a software receiver can dynamically adapt to a variety of signal environments, thereby enhancing its resistance to a wide range of signal degradations, including multipath effects, interference, and other disturbances. Furthermore, the use of a software-based methodology enables the smooth incorporation of additional features, including assistance for numerous constellations, including GPS, Galileo, GLONASS, and Indian regional navigation satellite system (IRNSS), in addition to augmentation systems designed to improve the accuracy and dependability of the system [5], [6]. Satellite navigation systems have been substantially advanced by GNSS software receivers, which regularly depend on host PCs and digital signal processors (DSPs). However, these solutions are restricted in their ability to be modified to accommodate a variety of circumstances [7]. The immobility of host PC-based receivers is a particularly noteworthy constraint. The fixed configurations of these receivers, which rely on the computational capabilities of tethered computers, restrict their applicability to tasks that demand portability and compactness.

In a similar vein, although receivers that employ DSP architecture have a more compact physical appearance, they might not possess the necessary adaptability to facilitate portability. The constraint presents difficulties when attempting to incorporate these receivers into portable devices or applications where weight and dimension restrictions are critical. Furthermore, it is imperative to consider the power consumption of both the host PC and DSP-based receivers. Power consumption can be significant due to the computational requirements of host PC-based systems and the need for efficient signal processing in DSP-based receivers [8]. This is particularly problematic in situations where power efficiency is critical, such as when operating in remote or resource-constrained areas or with battery-powered equipment. Moreover, in the case of these receivers, scalability and real-time performance pose further challenges. The computational capabilities of the connected PC frequently impact the scalability and real-time processing capabilities of host PC-based solutions [9]. On the contrary, DSP-based receivers may encounter difficulties when it comes to processing complex algorithmic structures or multiple satellite constellations in real time, despite their proficiency in signal processing. Furthermore, external apparatus, including specialised interfaces, antennas, or radio frequency (RF) front ends, is required for both varieties of receivers. This interdependence has the potential to undermine the system's overall dependability and maintenance, as well as escalate intricacy, expenses, and possible sites of failure [10].

Additionally, host PC-based systems share obstacles related to cost, intricacy, scalability, and maintenance with the implementation of DSP-based solutions. During maintenance or upgrades, adjustments or updates to the fundamental hardware are required, which makes it difficult to incorporate the most recent technological developments without requiring a substantial redesign. The constraints highlight the critical need for progress in receiver technologies that tackle issues including portability, energy efficiency, scalability, instantaneous functionality, and economical pricing. To address these limitations, emerging technologies like software-defined radio (SDR) and system on chip (SoC) solutions provide alternatives that are compact, energy-efficient, and highly customisable for GNSS signal processing and navigation applications.

## 2. RELATED WORKS

Grenier *et al.* [11] proposed the creation of an open-source research platform with the objective of delivering reference data and shared software that are customised to suit the unique requirements of individual users and environments. Wang *et al.* [12] devised smart-precise point positioning (PPP), a method that adapts the PPP algorithm, originally designed for geodetic receivers. This modification enables cost-effective devices to attain optimal positioning accuracy. Smart-PPP utilises a model that can integrate single-frequency and dual-frequency observations from tracked satellites. The devices designate distinct receiver clock terms for the phase measurements of each signal to rectify any potential discrepancy between the code and carrier phases. Additionally, adding ionospheric pseudo-observations to the PPP model makes it work better by making it very hard to guess oblique ionospheric delays.

According to Motella *et al.* [13], the procedure for integrating software routines that facilitate the operation of an open service navigation message authentication (OSNMA) receiver with real-time GPS/Galileo capabilities is comprehensively described. In addition to providing a comprehensive outline of the development endeavour, the paper also conveys insightful observations regarding implementation. Conversely, Ferreira *et al.* [14] presents an innovative methodology that effectively overcomes constraints and seamlessly incorporates the functionalities of RTKRCV into reactive oxygen species (ROS). To achieve seamless integration with pre-existing frameworks, the integration process adeptly integrates novel functionalities such as ROS publication and administration through a ROS service. Furthermore, solution consistency is substantially improved with the implementation of an innovative observation synchronisation method, specifically in the moving-baseline placement mode. Practical application instances are incorporated into the paper to emphasise the benefits of the proposed package.

Nie *et al.* [15] combined dual-frequency ionospheric-free code and phase data with single-frequency ionosphere-corrected code measurements and accurate ionospheric products to develop a new location determination method. Satellites with the second civil signal had ionospheric-free code and phase combinations, while all identified satellites had single-frequency ionosphere-corrected code measurements. Wang *et al.* [16] presented a compute unified device architecture (CUDA)-based GPU-based chip-shape correlator architecture that computes chip transitions and multiple correlator values via signal compression. An SDR-based A-GNSS receiver that can analyse multiple GNSS and regional navigation satellite systems (RNSS) signals across frequency bands was shown in [17]. Its modular construction makes this receiver flexible and expandable. For real-time applications, A-GNSS server software provides satellite ephemeris data quickly. Due to RF front-end sample bandwidth limits, many SDRs processed multi-GNSS/RNSS multi-frequency signals simultaneously. To avoid RF data overflow, memory buffer management was improved. The suggested SDR-A-GNSS receiver used concurrent NVIDIA CUDA GPUs and CPU threads for signal gathering and analysis. Kumar and Paidimarry [18] suggested using the USRP N210 kit to efficiently gather real-time GPS data. A novel data management queue thread technique was presented to ensure receiver calibration with accurate specifications to address data overflow and underflow difficulties.

Lapin *et al.* [19] introduced system for tasking and real-time exploitation (STARE), a real-time software receiver for location, for fifth generation (5G) new radio (NR) and long-term evolution (LTE) cellular downlink signals. STARE immediately interfaces with the SDR to process signals in real time without storing data on a disc. It allows multiple-channel SDRs and multiple tracking channels that acquire and track signals separately. To avoid overstating channel orders and resulting in inaccurate delays and phase estimates, the tracking channel uses a route selection criterion based on the signal-to-noise ratio (SNR) of the earliest path during acquisition. Bahadur and Nohutcu [20] used code and carrier phase measurements to combine ultra-rapid technology with conventional approaches for real-time single-frequency multi-GNSS localization. The project tested single-receiver single-frequency localization utilising the ionosphere-free code-carrier combination.

Srinu and Parayitam [21] examined the positional accuracy of GPS and IRNSS signals in the L5/S bands and the GPS software receiver changes needed to accept IRNSS. It summarises NavICSR's acquisition, tracking, and data decoding methods. MATLAB was used to construct a software application at Osmania University's NERTU Department [22]. Centre Tecnològic de Telecomunicacions de Catalunya (CTTC's) GNSS-SDR has a single module for post-processing and real-time use. A cheap RTL-SDR front-end receives the 1A satellite's digitised NavIC L5 IF signal. Verifying decoded potential fishing zone (PFZ) information against Indian National Centre for Ocean Information Services messages evaluates the messaging receiver. This receiver can evaluate and monitor message services in real time and optimise their use in corporate and civilian settings. Nicola *et al.* [23] discusses the unusual integration of chimaera protocol verification into a GPS software receiver. Chimaera uses authentication chips in the propagating code. These entities are cryptographically constructed using a key from the navigation message's digital signature or another source. Chimaera marker verification is implemented by comparing stored analog-to-digital converter (ADC) signal samples to receiver marker samples in this investigation. The article also details the L1C signal software tool, including the chimaera improvement. Since live signals are absent, a suitable software generator is needed to test the receiver.

## 3. METHOD

This section presents a design proposal for a SoC-based real-time GNSS software receiver. By capitalising on an open-source GNSS SDR framework that was initially designed for personal computers, this advancement seeks to optimise the incorporation of diverse RF front-ends and expedite the development of the receiver chain for navigation solutions. The chosen framework, which originates from the open-source GNSS SDR group, is notable for its straightforwardness, error-free operation, and code reusability. Its adaptability allows for the creation of optimal executions that are compatible with a wide range of hardware platforms and operating systems. Furthermore, this framework specifically designs dedicated runtime signal processing elements for the implementation of software radio applications.

PCs can function with software receivers and modify them to support a wide range of RF front ends. This system processes raw samples, either in real-time or from offline-stored files. Several signal processing components compose the framework, such as the source, conditioner, channel, acquisition, tracking, telemetry decoder, observables, and position, velocity, and time (PVT). Supplying an uninterrupted flow of unprocessed data, the signal source module operates via an RF front-end or stored file. As an intermediary, the signal conditioner establishes a connection between the signal source and the channel. The signal conditioner primarily executes two functions. The signal conditioner filters the signal from the signal source and then downconverts it according to the channel's specifications. The channel incorporates GNSS software baseband processing techniques, including decoding, tracking, and acquisition of navigation data. Each channel executes the baseband processing algorithms for a specific satellite, and the user can configure the number of parallel channels through software configuration. Following the collection of synchronised data from multiple signal processing blocks by the observable block, the system generates crucial GNSS metrics such as simulated range, carrier phase, and doppler shift. The high computational complexity of the software receiver on a host PC significantly limits its deployment. Determining the initial position of the software receiver running on the host PC requires a significant investment of effort. We advise against these constraints and suggest the implementation of a SoC-based software receiver. The software receiver, which is integrated into the SoC platform, effectively acquires real-time GNSS signals by utilising the USRP N210 RF front-end module. The Zynq 7000 SoC board implements the algorithms for processing the GPS baseband. Figure 1 depicts the block design of a SoC-based GNSS software receiver.

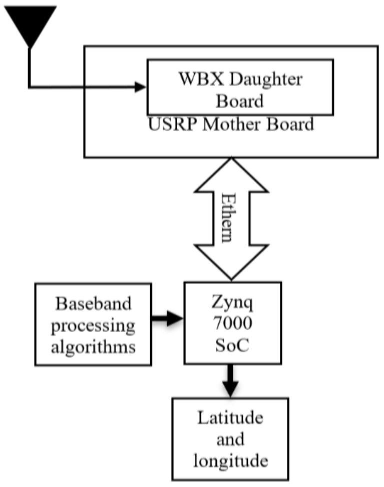


Figure 1. GNSS SDR in SoC

The critical function of the GPS active antenna in capturing real-time GPS signals is depicted in Figure 1. The receiver end utilises the wide bandwidth transceiver (WBX) daughterboard to enable frequency band selection, encompassing a frequency range of 50 megahertz to 2 gigahertz. This enables the effective recording of all GNSS signals. An ethernet connection is used to construct a direct interface between the USRP N210 and the Zynq 7000 SoC; however, the necessary drivers for integration are not present. In order to address this disparity, a tailored ethernet driver has been created [24]. In order to optimise the process of data management, the novel data management queue thread methodology is implemented. The Zynq 7000 SoC processor facilitates the execution of GPS baseband processing methods, which are responsible for determining the user's location.

### 3.1. Global navigation satellite system software-defined radio baseband processing algorithms

The foundation of GNSS receivers consists of baseband processing algorithms, which are responsible for decoding and extracting critical data from satellite signals. Algorithms such as these are crucial in the management of signal acquisition, tracking, and data demodulation, thereby guaranteeing precise positioning in the face of external interference. In the initial phase of signal acquisition, the receiver identifies and secures satellite signals. Sophisticated algorithms can efficiently synchronise with satellites across a wide range of frequencies and codes, even in extremely crowded urban areas or in conditions of feeble signal strength. Following their acquisition, surveillance algorithms are activated to monitor and subsequently latch onto these signals. By adapting dynamically to atmospheric or physical signal distortions, these systems guarantee the ongoing and accurate surveillance of satellite trajectories. Furthermore, data demodulation methodologies are implemented in order to decipher these signals. These techniques exhibit the capability to decipher encrypted navigation data, which in turn reveals crucial information, including clock corrections, satellite orbits, and other essential location parameters. Three distinct components comprise this framework: acquisition, tracking, and data decoding.

#### 3.1.1. Acquisition

Acquisition involves a systematic inquiry that seeks to identify and localise pseudo-random-noise (PRN) codes in addition to the code phase of GNSS signals that have been detected. This operation is executed utilising the acquire function and functions as an essential preliminary stage to commence the monitoring procedure. The level of accuracy needed is set by the chip width of the discriminators used by the code tracking loop and the bandwidth of the carrier tracking loop working together. The search technique investigates every possible combination by utilising initial estimations. This intricate process requires substantial computational resources and the provisional retention of data for several milliseconds in random-access memory (RAM). It is critical to finish the acquisition phase prior to initiating the tracking phase, as this establishes its primacy in the GNSS measurement procedure. To estimate the visible satellites in incoming signals, Magiera *et al.* [25] have identified three distinct acquisition methods: parallel code phase search (PCPS), parallel frequency search (PFS), and serial search.

We identify the serial search as the most direct method for performing arithmetic operations; it is only pertinent to fundamental operations such as addition, multiplication, and square. The sequential search procedure, on the other hand, results in an increase in computational time. The fourier transform is utilised by the two remaining methods to facilitate the conversion of signals from the time domain to the frequency domain. As a result, a more thorough investigation of the frequency spectrum becomes viable. We conduct both a coarse and a refined search during the acquisition process. In order to determine GNSS codes and provide an approximate doppler frequency, the former functions in conjunction with the PCPS. This preliminary investigation identifies the code phase with exceptional accuracy. Conversely, when combined with the PFS, the fine search yields a result that closely matches the detected GNSS signal's coarse doppler frequency.

Figure 2 depicts the PCPS acquisition method, renowned for its rapidity and effectiveness in obtaining GNSS signals. Initially, a local oscillator divides the incoming signal into two constituent parts, denoted as I and Q. We then implement a fourier transform (FFT) procedure on these components. Next, we process the FFT result by multiplying it with the complex conjugate of the local code sequence. This multiplication operation creates an additional FFT process. The second FFT's output illustrates the correlation between the incoming signal and the local code sequence. Next, peaks in the correlation output are scrutinised as they function as indicators of the presence of the signal. The PCPS acquisition method distinguishes itself from traditional sequential search techniques by virtue of its exceptional capability to search all prospective code phases simultaneously. Concurrent searching not only improves efficacy, but it also significantly reduces acquisition time.

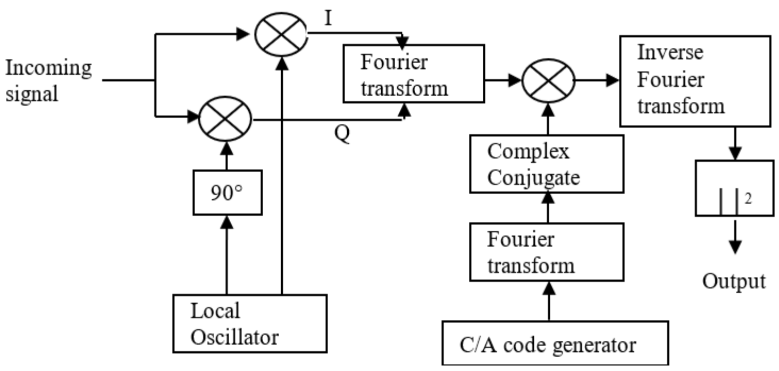


Figure 2. PCPS acquisition method

### 3.1.2. Tracking

Tracking is crucial in GNSS software receivers because it synchronises with satellite signals to pinpoint the user's location. This level includes carrier and code tracking, which are crucial. Carrier tracking accurately determines satellite broadcast phase and frequency to maintain synchronisation with the satellite's carrier signal. This approach requires the highest accuracy to correct for doppler shifts caused by satellite-receiver movement. We constantly improve receiver computations of carrier phase, code phase, and other positioning parameters. These processed data enable receiver position, timing, and velocity determination. During tracking, the receiver uses advanced algorithms and signal processing to monitor multiple satellite signals simultaneously. The receiver iteratively refines its estimations to improve location data accuracy and provide a reliable navigation solution. Figure 3 block diagram depicts the configuration of a carrier and code tracking loop system, which is a typical element of a GNSS software receiver. GNSS terminology frequently refers to the code tracking loop as a delay lock loop (DLL). On the other hand, GNSS terminology designates the carrier tracking loop as a phase lock loop (PLL).

The code tracking loop's critical function is to monitor the coding phase of an incoming GNSS signal with extreme precision. The phase in question represents the difference between the incoming signal's PRN code and the local PRN code generated by the receiver. Correlating the incoming signal with three locally generated PRN codes-early, prompt, and late-is its mode of operation. We deliberately offset the "early" and "late" codes by a small margin in relation to the prompt code. The carrier tracking loop mostly adjusts to the incoming GNSS signal's carrier frequency. The carrier frequency is the sine wave modulating the GNSS signal. To function, this loop compares the incoming signal's carrier frequency phase to the locally generated carrier frequency. It adjusts the locally generated carrier frequency via a phase difference.

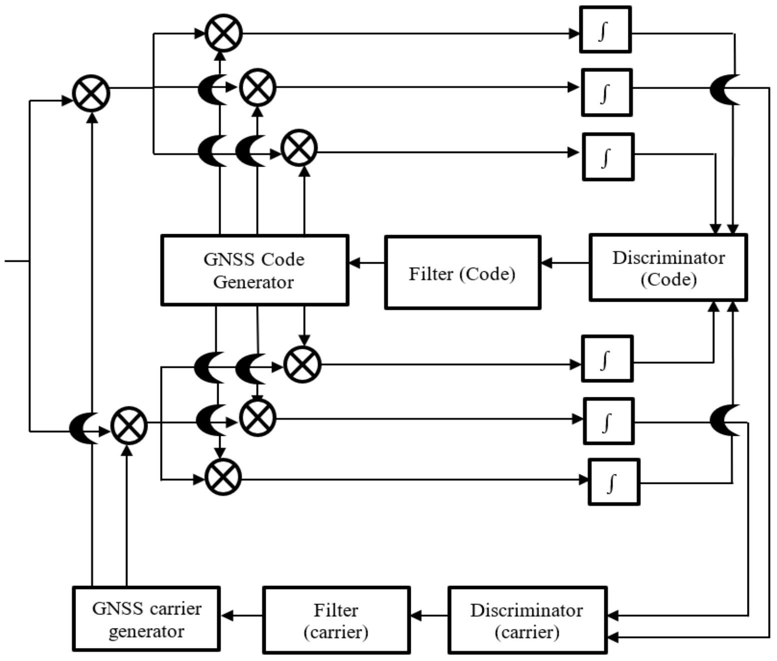


Figure 3. Code and carrier loops in tracking

## 4. RESULTS AND DISCUSSION

SDR can be implemented on DSPs, field-programmable gate arrays (FPGAs), SoCs, and host PCs. In these platforms, SoC is more flexible. SoC-implemented real-time GNSS SDR is our research focus. The Zynq 7000 programmable SoC was used to create GNSS baseband processing algorithms first. The USRP N210 kit and ADFMCOMMS 3-EBZ were used as RF front-ends. The chosen SoC platform is multi-core for flexibility and cost-effectiveness. Baseband processing algorithms from the GNU radio open-source programme helped us comprehend basic operations. GNU radio offers many communication toolboxes for Linux (Ubuntu). Adding the GNU Radio tool and USRP hardware driver (UHD) and analogue device RF drivers allowed control of both RF front ends. A brief description of the Zynq 7000 SoC and RF front ends follows. The Xilinx Zynq-7000 extensible processing platform powers the ZedBoard, a development and evaluation platform. The Zynq-7000 EPP's dual-core cortex-A9 processing system (PS) and 85,000 series-7 programmable logic (PL) cells make it versatile. With its onboard peripherals and expandability options, the ZedBoard suits designers of all levels. Figure 4 shows the multi-core SDR design. The USRP kit has two antennas that can receive and send signals. These antennas catch zero-IF signals around the GPS L1 frequency of 1575.42 MHz. GPS L1 signals typically have a 10 MHz bandwidth. The daughter boards in USRP kits digitise signals after reception. The host computer's hard disc stores the digital signal. For USRP kit operation, GNU Radio software and the UHD driver are required. A host computer running Ubuntu Linux runs this programme. The Zynq 7000 SoC target device connects directly to the USRP N210 RF front end. USRP RF front-end management uses a bespoke Linux operating system that supports innovative baseband processing methods. The Zynq SoC incorporates these changes. Table 1 lists experimental acquisition and tracking configuration options.

The ADFMCOMMS3-EBZ RF front end and the Zynq 7000 SoC establish a communication link through a standard parallel 110 interface, allowing for the transmission of high-speed sampled data at an estimated 123 complex MSPS. Additionally, the serial peripheral interface (SPI) serves the functions of configuration, management, and surveillance. Figure 5 depicts the integration of the Zynq 7000 SoC board with the ADFMCOMMS3-EBZ interface.

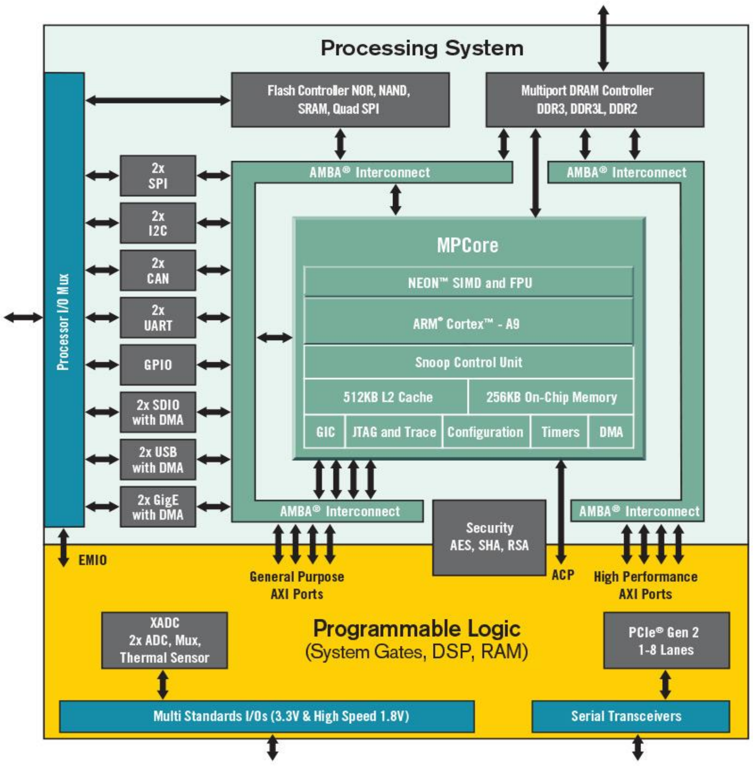


Figure 4. Zynq 7000 SoC architecture

Table 1. GNSS SDR receiver configuration

| S. No | Parameter | Configuration value | |
|---|---|---|---|
| | | GPS | IRNSS |
| 1 | Centre frequency | 1575.42MHz | 1176.45 MHz |
| 2 | Sampling frequency | 20MHz | 20MHz |
| 3 | Gain | 60dB | 60dB |
| 4 | No of channels | 12 | 8 |
| 5 | Acqustion threshold | 0.2 | 0.1 |
| 6 | Freqency step | 300Hz | 300Hz |
| 7 | PLL bandwidth | 40Hz | 30Hz |
| 8 | DLL bandwidth | 6Hz | 4Hz |

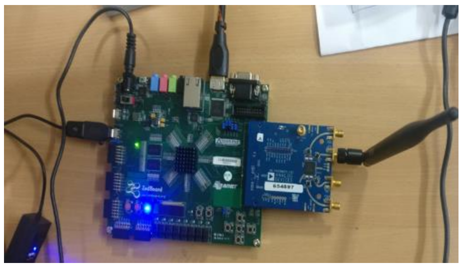
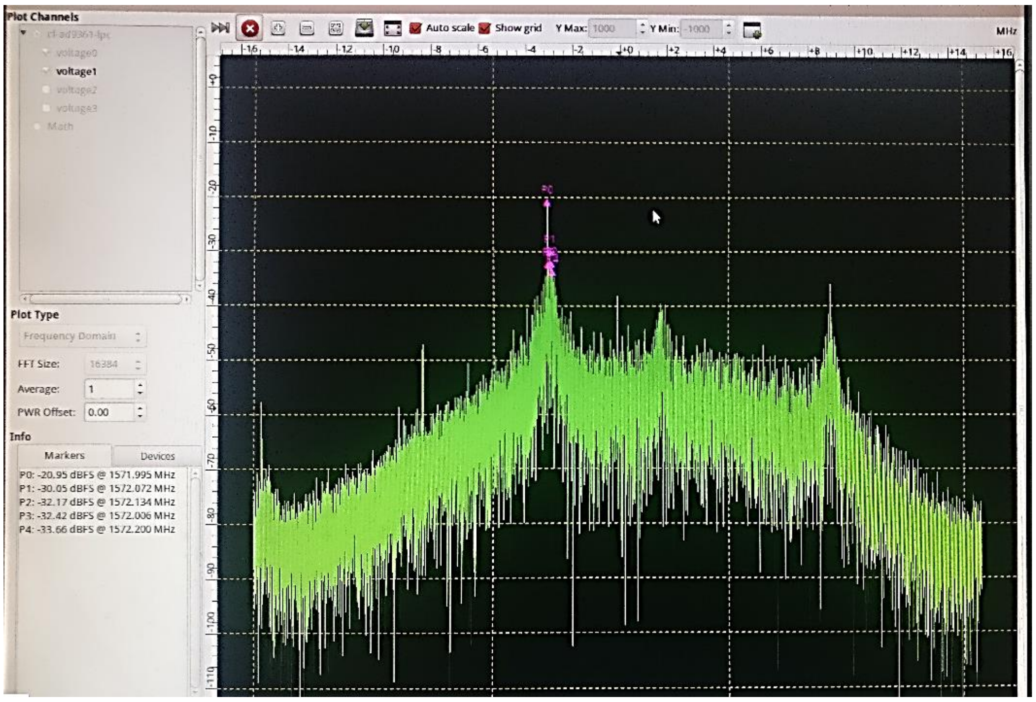


Figure 5. Zynq 7000 SoC with ADFMCOMMS3-EBZ

The system uses internal static random access memory (SRAM) to process SPI samples. FPGA topologies allocate blocks for IQ correction, up/down conversion, decimation/interpolation, and higher-order computation. One A9 processor handles modulation and packet processing, while the other controls transmit and receive signals and data flow. A fast SoC AXI bus connects CPU cores with FPGA fabric. The system uses the ADFMCOMMS3-EBZ framework and works at 1575.42 MHz, the GPS L1 frequency band. The signal bandwidth is 10 MHz, and the sampling rate is 30 MSPS. Hardware gain is 71 dB to boost the signal. For reliable connectivity, the system receives satellite signals via a Vert 900 antenna. Figure 6 shows the frequency spectrum.



Plot Type: Frequency Domain
FFT Size: 16384
Average: 1
PWR Offset: 0.00

Info: Markers | Devices

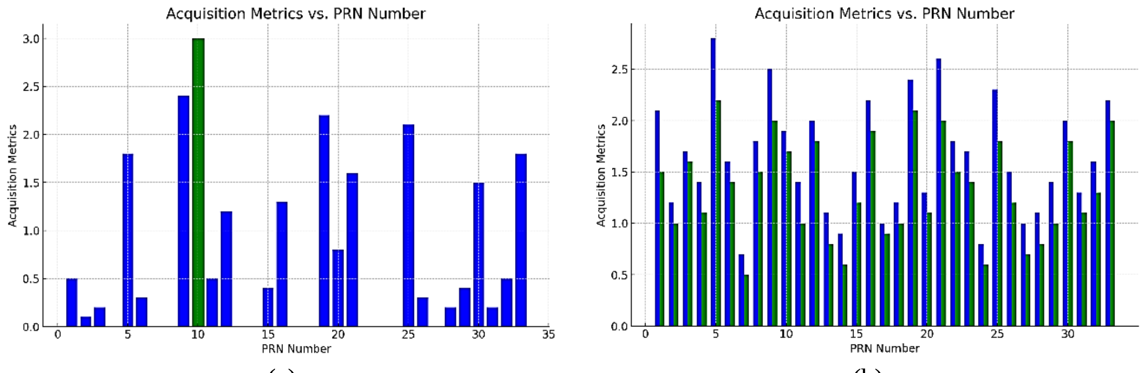
```
P0: -20.95 dBFS @ 1571.995 MHz
P1: -30.05 dBFS @ 1572.072 MHz
P2: -32.17 dBFS @ 1572.134 MHz
P3: -32.42 dBFS @ 1572.006 MHz
P4: -33.66 dBFS @ 1572.200 MHz
```

Figure 6. GPS L1 frequency spectrum

Peak markers mark real-time GPS signal peaks at frequencies of 1571.995 MHz, 1572.072 MHz, 1572.134 MHz, and 1572.20 MHz, with gains ranging from -20.95 to -33.66 dB. This study emphasises the importance of evaluating satellite data to determine the methodology's accuracy. To do this, our results are compared to those from the conventional PFS approach and validated using U-center software.

Figure 7(a) shows a histogram of conventional methodology acquisition data. The graph displays PRN numbers for each satellite on the horizontal axis, as well as their optimal metric values on the vertical axis. A visibility threshold of 2 was selected during the experiment to match the signal content in the received data. Satellites with a maximum metric value below this threshold are marked blue. Figure 7(a) shows that no satellite's peak metric value exceeds the threshold. Insufficient samples for signal processing cause an incoming signal's carrier frequency to vary from its original frequency, resulting in data loss.

Figure 7(b) shows the results of the acquisition algorithm after data collection, using our proposed method of data control via the queue thread. Satellites that demonstrate peak metrics that exceed a predetermined threshold are classified as visible and denoted by the colour green highlighting. Satellites 1, 11, 13, 17, and 30 have been identified as visible. The u-center application, compatible with both mobile devices and personal computers, simplifies the administration of the u-blox GPS sensors. Figure 8 illustrates how to execute the u-center application on a mobile device to observe visible satellites.



(a) (b)

Figure 7. Acquisition results of standard and proposed PCPS method: (a) regular method and (b) proposed method

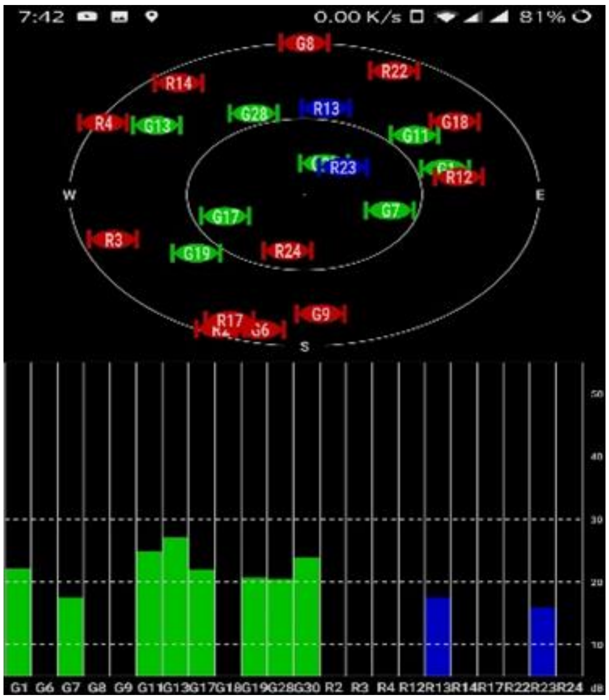

7:42
0.00 K/s 81%
G8
R22
R14
R13
R4
G13
G28
G18
G11
R23
R12
W
E
G17
G7
R3
G19
R24
R17
G6
G9
S
G1 G6 G7 G8 G9 G11 G13 G17 G18 G19 G28 G30 R2 R3 R4 R12 R13 R14 R17 R22 R23 R24 dB
50
40
30
20
10

Figure 8. Detected satellites from U-center software

Under stationary conditions, the study evaluates the precision of receiver location estimation using a SoC-based method. This configuration features the active GPS antenna positioned atop the tallest point of a structure, thereby guaranteeing an atmosphere devoid of obstructions. Preloaded baseband processing algorithms are utilised to process the incoming signal on the SoC board. After performing the computation of PVT, the OLED panel exhibits the resultant latitude and longitude coordinates. The displayed latitude and longitude coordinates are 78.51773 and 17.40774, respectively. The SoC-based receiver calculates the initial location (latitude and longitude) in an average of ten seconds. We undertake a comparative examination between the SoC-based receiver and a commercially accessible Ublox receiver to authenticate these results.

## 5. CONCLUSION

A GNSS-SDR receiver has been successfully developed using a USRP RF front end. The acquisition of GNSS data in real-time was accomplished by utilising GNU-radio, a programme that is both free and open-source. The GNSS baseband processing methods were modified to allow them to be executed on a SoC target device. This modification enabled the capture and handling of GPS-L1 signals, making a noteworthy contribution to this effort. During the procedure, incoming signals were processed using preloaded baseband processing algorithms that were built into the SoC chip. After the PVT computation was finished, the resulting latitude and longitude values were shown on an OLED panel. A comparative analysis was undertaken to verify the correctness of the experimental results by comparing the SoC-based receiver with an Ublox receiver.

## REFERENCES

[1] R. B. Langley, P. J. G. Teunissen, and O. Montenbruck, "Introduction to GNSS," in *Springer Handbook of Global Navigation Satellite Systems*, Cham: Springer International Publishing, 2017, pp. 3–23, doi: 10.1007/978-3-319-42928-1_1.

[2] A. Kumar, S. Kumar, P. Lal, P. Saikia, P. K. Srivastava, and G. P. Petropoulos, "Introduction to GPS/GNSS technology," in *GPS and GNSS Technology in Geosciences*, Elsevier, 2021, pp. 3–20, doi: 10.1016/B978-0-12-818617-6.00001-9.

[3] M. J. Unwin *et al.*, "An introduction to the HydroGNSS GNSS reflectometry remote sensing mission," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 6987–6999, 2021, doi: 10.1109/JSTARS.2021.3089550.

[4] Z. Yao and M. Lu, *Next-generation GNSS signal design: theories, principles and technologies*, Singapore: Springer, 2021, doi: 10.1007/978-981-15-5799-6.

[5] R. Gautam, V. Chaudhary, and S. Gajjar, "GUI development of IRNSS receiver," in *Emerging Technology Trends in Electronics, Communication and Networking*, Singapore: Springer, 2023, pp. 119–127, doi: 10.1007/978-981-19-6737-5_11.

[6] P. Rakshit, S. Dan, B. Das, A. Santra, S. Mahato, and A. Bose, "Compact, low-cost, single-frequency NavIC receiver development," in *2020 URSI Regional Conference on Radio Science*, Feb. 2020, pp. 1–4, doi: 10.23919/URSIRCRS49211.2020.9113575.

[7] D. E. -Roca *et al.*, "GNSS user technology: state-of-the-art and future trends," *IEEE Access*, vol. 10, pp. 39939–39968, 2022, doi: 10.1109/ACCESS.2022.3165594.

[8] A. Grenier, E. S. Lohan, A. Ometov, and J. Nurmi, “A survey on low-power GNSS,” *IEEE Communications Surveys & Tutorials*, vol. 25, no. 3, pp. 1482–1509, 2023, doi: 10.1109/COMST.2023.3265841.
[9] P. Singh, J. Joshi, A. Dey, and N. Sharma, “GNSS satellite selection-based on per-satellite parameters using deep learning,” *IETE Journal of Research*, vol. 70, no. 1, pp. 46–57, Jan. 2024, doi: 10.1080/03772063.2022.2121768.
[10] O. Diouri, A. Gaga, H. Ouanan, S. Senhaji, S. Faquir, and M. O. Jamil, “Comparison study of hardware architectures performance between FPGA and DSP processors for implementing digital signal processing algorithms: application of FIR digital filter,” *Results in Engineering*, vol. 16, Dec. 2022, doi: 10.1016/j.rineng.2022.100639.
[11] A. Grenier, E. S. Lohan, A. Ometov, and J. Nurmi, “An open-source software-defined receiver for GNSS algorithms benchmarking,” in *2022 14th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, Oct. 2022, pp. 31–38, doi: 10.1109/ICUMT57764.2022.9943489.
[12] L. Wang, Z. Li, N. Wang, and Z. Wang, “Real-time GNSS precise point positioning for low-cost smart devices,” *GPS Solutions*, vol. 25, no. 2, pp. 1–13, Apr. 2021, doi: 10.1007/s10291-021-01106-1.
[13] B. Motella, M. T. Gamba, and M. Nicola, “A real-time OSNMA-ready software receiver,” in *Proceedings of the 2020 International Technical Meeting of The Institute of Navigation*, Feb. 2020, pp. 979–991, doi: 10.33012/2020.17191.
[14] A. Ferreira, B. Matias, J. Almeida, and E. Silva, “Real-time GNSS precise positioning: RTKLIB for ROS,” *International Journal of Advanced Robotic Systems*, vol. 17, no. 3, May 2020, doi: 10.1177/1729881420904526.
[15] Z. Nie, F. Liu, and Y. Gao, “Real-time precise point positioning with a low-cost dual-frequency GNSS device,” *GPS Solutions*, vol. 24, pp. 1–11, Jan. 2020, doi: 10.1007/s10291-019-0922-3.
[16] C. Wang, X. Wang, X. Cui, G. Liu, and M. Lu, “Efficient chip-shape correlator implementation on a GPU-based real-time GNSS SDR receiver,” *GPS Solutions*, vol. 26, no. 4, Oct. 2022, doi: 10.1007/s10291-022-01332-1.
[17] B. S. Nayak, K. K. Naik, O. Ojjela, and S. Pal, “GPS receiver simplification for low cost applications and multipath mitigation analysis on SDR based re configurable software receiver,” *Defence Science Journal*, vol. 73, no. 6, pp. 699–711, Nov. 2023, doi: 10.14429/dsj.73.19033.
[18] B. P. Kumar and C. S. Paidimarry, “Improved real time GPS RF data capturing for GNSS SDR applications,” *Gyroscopy and Navigation*, vol. 11, no. 1, pp. 59–67, Jan. 2020, doi: 10.1134/S2075108720010083.
[19] I. Lapin, G. S. Granados, J. Samson, O. Renaudin, F. Zanier, and L. Ries, “STARE: real-time software receiver for LTE and 5G NR positioning and signal monitoring,” in *2022 10th Workshop on Satellite Navigation Technology (NAVITEC)*, Apr. 2022, pp. 1–11, doi: 10.1109/NAVITEC53682.2022.9847544.
[20] B. Bahadur and M. Nohutcu, “Real-time single-frequency multi-GNSS positioning with ultra-rapid products,” *Measurement Science and Technology*, vol. 32, no. 1, Jan. 2021, doi: 10.1088/1361-6501/abab22.
[21] C. Srinu and L. Parayitam, “A post processing based IRNSS/NavIC software receiver for analysis and development of new algorithms and signals,” in *Proceedings of the 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020)*, Oct. 2020, pp. 2822–2835, doi: 10.33012/2020.17597.
[22] R. R. D, C. Srinu, and L. Parayitam, “Low-cost real-time software receiver for IRNSS/NavIC short broadcast messaging services,” in *34th International Technical Meeting of the Satellite Division of The Institute of Navigation*, Oct. 2021, pp. 549–558, doi: 10.33012/2021.18144.
[23] M. Nicola, B. Motella, and M. T. Gamba, “GPS chimera: a software receiver implementation,” in *34th International Technical Meeting of the Satellite Division of the Institute of Navigation*, Oct. 2021, pp. 4264–4273, doi: 10.33012/2021.18127.
[24] C. Diouf, G. J. M. Janssen, H. Dun, T. Kazaz, and C. C. J. M. Tiberius, “A USRP-based testbed for wideband ranging and positioning signal acquisition,” *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–15, 2021, doi: 10.1109/TIM.2021.3065449.
[25] W. Magiera *et al.*, “Accuracy of code GNSS receivers under various conditions,” *Remote Sensing*, vol. 14, no. 11, May 2022, doi: 10.3390/rs14112615.

## BIOGRAPHIES OF AUTHORS

**Chetna Devi Kh** received the B.Tech. degree in Electronics and Communication engineering from Anna University, Chennai, India, in 2012; the M.Tech. degree in VLSI Design and Embedded System from PESIT under VTU, Belgaum, in 2014. Currently she is pursuing her Ph.D. from Jain University. Her research interests are VLSI design and software defined radio. She can be contacted at email: choisiwon.kh@gmail.com.

**Malode Vishwanatha Panduranga Rao** obtained his Ph.D. degree in computer science from National Institute of Technology Karnataka, Mangalore, India. He has completed a Master of Technology in computer science and Bachelor of Engineering in Electronics and communication Engg. He is currently working as Professor in Jain (Deemed to be University) Bengaluru, India. His research interests are in the field of real-time and embedded systems - internet of things. He has published various research papers in journal and conferences across India, also in the IEEE international conference in Okinawa, Japan (visited) 2008. He has authored two reference books on Linux Internals. He is the life member of Indian Society for Technical Education and IAENG. Now from past three years, he has published 12 indian patents and three patents are stepping towards grant status. One research scholar under his guidance was awarded a Ph.D. degree. He can be contacted at email: r.panduranga@jainuniversity.ac.in.