International Journal of Scientific Research and Engineering Development-– Volume 7 Issue 3, May-June 2024
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

AMPLITUDE MODULATION AND DEMODULATION USING FPGA(VIVADO), GNURADIO, SDR AND SIMULATED GPS SIGNAL AND GOT IT LOCKED TO GPS USER RECEIVER BY SPOOFING THE SIGNALS Using SDR

Suresh.D*, M. Noorjahan**, Vengatesan.M***, Sethuraman. P****

*Scientist Engg- "SF" Navigation systems Area, ISTRAC , IRSO, BANGALORE

**Assistant Professor, Electronics & Communication Engineering, DhanalakshmiSrinivasan Engineering College (Autonomous), Perambalur, Tamil Nadu.

Email:*dsuresh@istrac.gov.in,**m.noorjahanme@gmail.com

****,****UG - Electronics & Communication Engineering, DhanalakshmiSrinivasan Engineering College(Autonomous), Perambalur, Tamil Nadu.

Email:*** vsan1509@gmail.com, ****sethu3656@gmail.com.

Abstract-The paper explores the integration of these technologies to create a comprehensive solution for amplitude modulation and demodulation in communication systems. FPGA, represented by the Xilinx Vivado design suite, is used to generate the carrier signal, perform the modulation and demodulation operations, and handle the digital signal processing.GNU Radio, an open-source software development toolkit, is utilized to simulate and test the modulation and demodulation algorithms before implementation on the FPGA. The SDR hardware, such as USRP or HackRF One, is employed to generate and process the analog signals, while the FPGA and GNU Radio handle the digital signal processing tasks.

The workflow involves designing the amplitude modulation and demodulation algorithms in Vivado, simulating and testing them using GNU Radio and the SDR hardware, and then implementing the FPGA-based system integrated with the SDR platform. This approach allows for a flexible and powerful implementation of amplitude modulation and demodulation, leveraging the strengths of FPGA hardware, GNU Radio software, and SDR platforms to create a comprehensive communication system. This work explores the simulation and spoofing of GPS signals using SDR technology. The objective is to generate simulated GPS signals and successfully lock them onto a GPS user receiver, effectively spoofing the receiver.

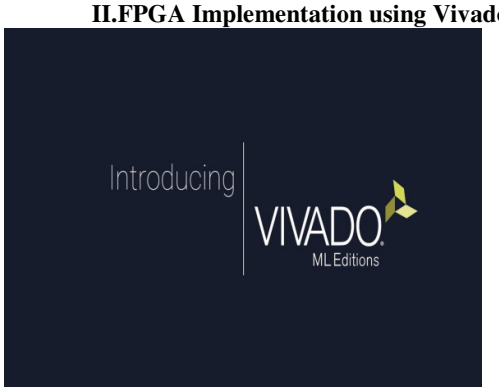
I.INTRODUCTION

Amplitude Modulation (AM) is a fundamental technique in communication systems used to transmit information over radio waves. It involves the AM transmitter, which generates an audio signal with a high-frequency carrier signal, and the AM receiver, which demodulates the AM signal to extract the original audio signal. The AM signal's amplitude varies according to the audio signal, and the carrier frequency is typically much higher than the audio frequency. The AM signal is then transmitted via an antenna.

The AM receiver then demodulates the AM signal, detects the envelope using a diode or rectifying circuit, and uses a low-pass filter to remove the high-frequency carrier, leaving behind the original audio signal. The demodulated audio signal can be expressed as m(t) = \frac{s(t)}{\cos(2\pi f_c t)} - 1.

FPGA implementation tools like Vivado and GNU Radio are used for designing and implementing digital circuits on FPGAs. Software-defined radios (SDRs) and signal processing applications are also used for flexible and programmable radio communication. Simulated GPS signals are used to test and validate receiver performance, while spoofing involves generating fake GPS signals to deceive a receiver. Achieving synchronization with a real GPS receiver is essential for accurate positioning.

II.FPGA Implementation using Vivado



Vivado is a design software developed by Xilinx for designing and implementing digital systems using FPGAs (Field-Programmable Gate Arrays) and adaptive SoC (System-on-Chips). It includes key components such as Design Entry, Synthesis, Place and Route, Verification/Simulation tools, and Implementation.

Design Entry allows users to create and edit HDL designs using either VHDL or Verilog, allowing them to

express their digital circuits at the high-level RTL (Register Transfer Level). During synthesis, Vivado translates your RTL code into a gate-level netlist, optimizing the design for area, speed, and power. Place and Route places synthesized logic elements onto the FPGA fabric, ensuring proper timing and resource utilization.

Verification and Simulation Tools include simulation capabilities to verify your design's functionality before implementation, allowing you to catch potential issues early in the design process. High-Level Synthesis (HLS) allows you to write your design in C/C++ and automatically generates RTL code, making it ideal for accelerating algorithmic designs. Power Estimation estimates power consumption during synthesis and implementation, helping you optimize power usage.

Vivado covers a wide range of Xilinx FPGAs and SoCs. The latest version of Vivado 2018.3 offers enhancements for various operating systems and device families, accelerated design integration, improved power estimation accuracy, video processing enhancements, high-level synthesis enhancements, enhanced debugging tools, security enhancements, automatic place & route of SLR crossings, improved visualization for DFX floorplans, Tandem+DFX Support, Vivado Simulator VCD Support for System C Users, and device support updates for various FPGA families.

Creating a new project in Vivado is straightforward, with steps such as opening the Vivado installer, extracting the downloaded package, running the installation application, choosing desired components, and completing the installation process. Creating a new project involves setting the project name, selecting the target FPGA device, importing VHDL source files, and configuring project settings.

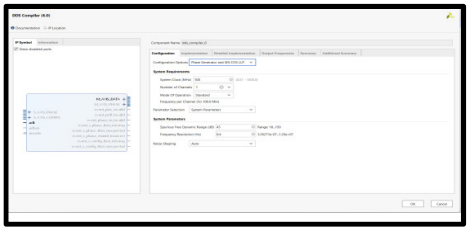
VHDL (VHSIC Hardware Description Language) is a powerful language for describing digital circuits and systems. Vivado is a comprehensive design suite that allows users to create, simulate, synthesize, and implement FPGA designs.

Vivado is a project-based design flow that involves creating models, adding constraints, simulating, synthesizing, implementing, and verifying functionality. It provides a project-based design flow, which includes HDL coding techniques, constraint use, simulation, synthesis and implementation, Microblaze Soft Core Processor creation, and Tcl Commands.

HDL coding techniques include writing VHDL code following proper conventions, making good pin assignments, using Xilinx Design Constraints (XDC) files, and adding additional constraints using Tcl scripting within Vivado. Simulation is used to verify design behavior, and designs can be analyzed using the Schematic viewer and Hierarchical viewer.

Synthesizing and implementation involve generating a netlist and mapping the design onto target FPGA resources. Users can customize their design by adding modules like UART for communication. Tcl Commands are familiarized with for generating a Microblaze processor.

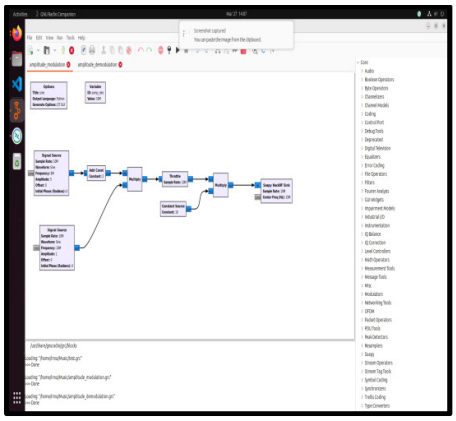
Add source files, IPs (Intellectual Property), simulation settings, and run behavioral simulation. Vivado does not directly provide individual gates like AND or NOT, but you can achieve similar functionality using specific blocks.



Utility Vector Logic (UVL) blocks operate on vector inputs and produce a vector output with logic applied to each bit. Available UVL gates include AND, OR, XOR, and NOT. URL gates reduce vector inputs to a single bit using AND, OR, and XOR operations.

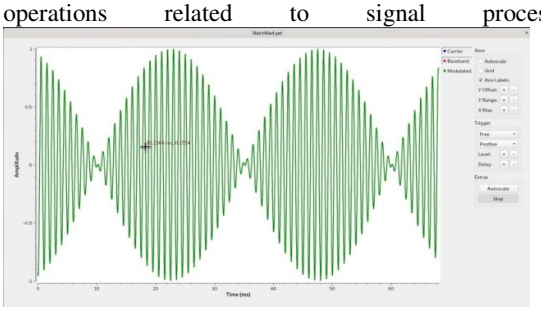
III.GNU RADIO

GNU Radio is a free and open-source software development toolkit that provides signal processing blocks for implementing software radios. It can be used with low-cost external RF hardware to create software-defined radios or without hardware in a simulation-like environment. GNU Radio is widely used in research, industry, academia, government, and hobbyist environments to support wireless communications research and real-world radio systems.



The GNU Radio Companion interface opens with a project titled "untitled" and includes four tabs: "amplitude modulation", "amplitude demodulation", "am_mod", and "untitled". The left side of the screen has a toolbar with icons for various actions like saving, undoing, redoing, zooming in/out, etc. The middle of the screen is a blank workspace area where blocks for building flowgraphs can be placed.

The GNU Radio Companion interface has four tabs: "amplitude modulation", "amplitude demodulation", "am_mod", and "untitled". The flowchart represents a signal processing pipeline for amplitude modulation, including the signal source, add const, multiply const, throttle, and WX GUI Scope Sink. The right-side menu labeled "Core" likely provides additional tools or operations related to signal processing.



The main feature of the image is a green waveform that oscillates between -1 and 1 in amplitude. The x-axis represents time in milliseconds (ms), while the y-axis represents amplitude. There are three distinct sections of the waveform: two areas with higher frequency oscillations and one area with lower frequency oscillations in the middle. A gray box on the right contains settings and adjustments for the displayed waveform, including options like "Carrier," "Baseband," and "Modulated," as well as controls for axes and triggering.

The amplitude demodulation block displays a waveform graphed against time, with the x-axis labeled "Time (ms)" and the y-axis ranging from -0.2 to 0.2. Grid lines at regular intervals on both axes aid in reading the graph. The waveform starts near zero, rises to a peak around 10 ms, falls below zero to a trough around 12 ms, rises back above zero to another peak at approximately 16 ms, and then declines towards zero again by 18 ms.

IV.AM MOD AND DEMOD Implementation using HACKRF ONE(SDR)

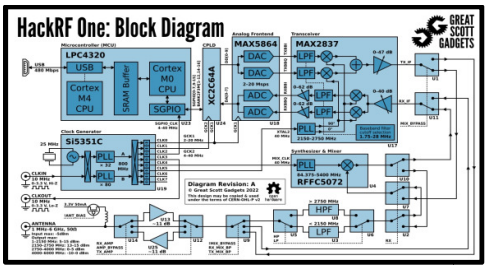
The HackRF One is a versatile Software Defined Radio (SDR) peripheral designed for a wide range of applications. It can receive and transmit radio signals across a broad frequency range, spanning from 1 MHz to 6 GHz, making it compatible with various wireless communication protocols such as Wi-Fi, Bluetooth, and GSM. The HackRF One is an open-source hardware platform, allowing developers and enthusiasts to modify, enhance, and contribute to its development.

Usage modes include USB peripheral, stand-alone operation, half-duplex transceiver capabilities, connectivity and ports, USB interface, power supply, and expansion and customization. The board includes pin headers for expansion, allowing users to add custom features or connect additional peripherals. It is a portable and versatile platform for experimenting with radio technologies.

Installation involves installing the necessary drivers and software for your operating system, with GNU Radio often used for signal processing. Advanced users can program HackRFOne using various programming languages like C/C++, Python, or even GNU Radio's graphical interface for signal processing.

Security research is another application of HackRF One, as it is often employed in security research for analyzing and assessing vulnerabilities of wireless systems, including Wi-Fi, Bluetooth, and other protocols. It can be used to intercept and analyze wireless communications, as well as to simulate attacks for testing purposes.

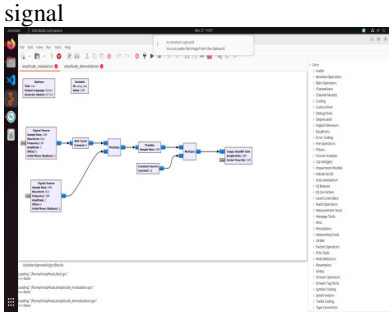
Community and resources around HackRF One offer tutorials, forums, and open-source projects to help users get started and advance their skills. The RESET button resets the microcontroller, leading to a new USB enumeration. The DFU button starts a loader USB DFU located in the microcontroller's ROM, which allows for restarting a HackRF One with damaged firmware. To call the DFU mode, press and hold the DFU button, press and release the RESET button or light the HackRF One, and release the DFU button.



In summary, the HackRF One is an excellent tool for radio enthusiasts, researchers, and developers who want to explore and experiment with wireless communication systems. Its open-source nature and wide frequency range make it a valuable addition to the SDR community.

The maximum power output (TX) varies according to the frequency of operation, with a maximum power of 30 MHz to 100 MHz and increasing as the frequency decreases. The output power is sufficient for performing live experiments at short distances or controlling an external amplifier. If connecting an external amplifier, an external bandpass filter must be used for frequency operation.

A signal processing pipeline for amplitude modulation and demodulation using SDR is represented by a flowchart that includes key components such as the signal source, add constant, multiply constant, and throttle. The WX GUI Scope Sink displays the signal waveform in a graphical scope, and the right-side menu labeled "Core" likely provides additional tools or operations related to signal processing.



The Amplitude Demodulation block using SDR also has a flowchart with measurements and settings, including Channel 1 (Ch1), Horizontal Scale, Trigger Level, Auto Trigger, Waveform Display, Bandwidth Limit, Sample Rate, and File Paths. The right-side menu labeled "Core" likely provides additional tools or operations related to signal processing.

The terminal output at the bottom shows file paths and executed commands. The Amplitude Demodulation block uses SDR to generate a sine wave with specified frequency, amplitude, and sampling rate, and the Waveform Display displays the waveform in a graphical scope.

V.SIMULATED GPS SIGNAL AND GOT IT LOCKED TO GPS USER RECEIVER BY SPOOFING THE SIGNALS

Simulated GPS signals are artificially generated signals that mimic the characteristics of real GPS signals transmitted by satellites. These simulated signals are used for various purposes, including testing, research, and training. Researchers and developers create simulated signals to evaluate the performance of GPS receivers, algorithms, and navigation systems without relying on actual satellite signals.

GPS spoofing involves intentionally transmitting fake GPS signals to deceive GPS receivers. This can manipulate the position, velocity, and timing information received by GPS devices. The attacker generates counterfeit GPS signals that appear authentic to the receiver, which may have altered coordinates or incorrect timing information. The GPS receiver, unaware of the deception, locks onto the spoofed signals, leading to an inaccurate position, leading to potential navigation errors.

Spoofing can have serious consequences, such as misleading autonomous vehicles, drones, or ships, disrupting critical infrastructure (e.g., power grids, financial systems) that rely on accurate GPS timing, and affecting military operations by misleading GPS-guided munitions or navigation systems. Countermeasures against GPS spoofing include receiver authentication, antenna arrays, signal monitoring, signal diversity, and jamming detection.

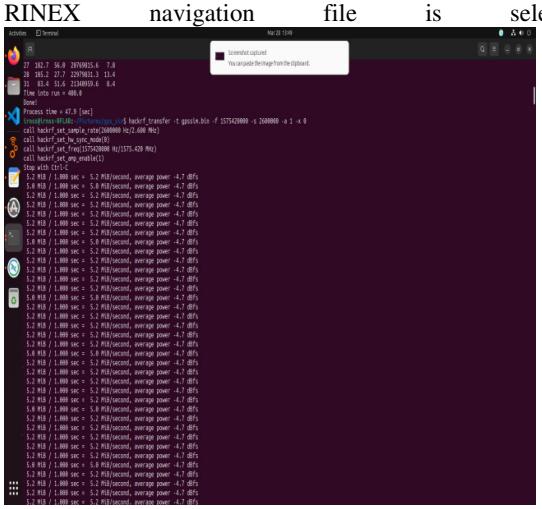
Real-world incidents of GPS spoofing have occurred, including a 2013 attack on a luxury yacht's navigation system in the Black Sea, researchers demonstrating successful spoofing attacks on drones, and some countries reporting incidents near sensitive locations, raising concerns about national security.

GPS-SDR-SIM generates GPS baseband signal data streams, which can be converted to RF using software-defined radio (SDR) platforms like ADALM-Pluto, bladeRF, HackRF, and USRP. To use bigger user motion files, the USER_MOTION_SIZE variable can be set to the maximum time of the user motion file in seconds. The setup suggests that the HackRF One receives signals (possibly simulated or real) and passes them through the attenuator before reaching the receiver.

The GPS signal file is generated using a user-defined trajectory in either a CSV file (containing the Earth-centered Earth-fixed (ECEF) user positions) or an NMEA GGA stream. The sampling rate of the user motion has to be 10Hz, and the user can also assign a static location directly through the command line. The GPS broadcast ephemeris file (brdc) is a merge of individual site navigation files into one, and these files are used to generate the simulated pseudo range and Doppler for the

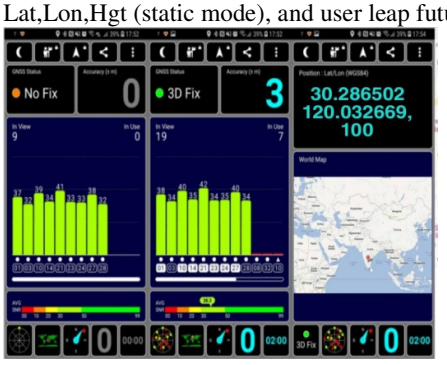
GPS satellites in view.

The bladeRF and ADALM-Pluto command line interface requires I/Q pairs stored as signed 16-bit integers, while the hackrf_transfer and gps-sdr-sim-uhd.py support signed bytes. The simulation start time can be specified if the corresponding set of ephemerides is available, otherwise the first time of ephemeris in the RINEX navigation file is selected.



The maximum simulation duration time is defined by USER_MOTION_SIZE to prevent the output file from getting too large. The output file size can be reduced by using the "-b 1" option to store four 1-bit I/Q samples into a single byte.

The gps-sdr-sim [options] option allows users to specify the RINEX navigation file for GPS ephemerides, user motion files in ECEF x, y, z format, lat, lon, height format, NMEA GGA stream, ECEF X,Y,Z in meters, Lat,Lon,Hgt (static mode), and user leap future event.



OUTPUT OF GNSS SPOOFING

The following is a list of commands and outputs related to various GPS scenarios:

1. BladeRF: Loading a simulated GPS signal file, "gpssim.bin", into the BladeRF for playback.
2. HackRF: Output data must be 8-bit signed I/Q samples.
3. LimSDR: Using a 1 Msps 1-bit file to get full BaseBand dynamic and low RF power.
4. ADALM-Pluto (PlutoSDR): Having its network interface up and running and accessible via "pluto.local" by default.
5. ADALM-PlutoSDR: Default settings for the ADALM-Pluto device.
6. PlumtoSDR: Default settings for the PlumtoSDR device.

VI.CONCLUSION

FPGA hardware acceleration Effective real-time AM modulation on the FPGA is made possible by Vivado. Software Flexibility (GNU Radio) offers an easy-to-use platform for AM process design and simulation. Functionality in the Real World (SDR) SDRs make it easier to transmit and receive AM signals in the real world. The creation of personalized GPS signals for controlled receiver behavior testing is made possible by the controlled environment (SDR). An analysis of vulnerabilities Finding weaknesses in GPS receivers is facilitated by simulating altered or weak signals. Better Receiver Design: The creation of more reliable GPS receivers can be guided by the insights gained from spoofing tests.

REFERENCES

[1] B. P. Lathi, "Modern Digital and Analog Communication Systems," Oxford University Press, 2010.

[2] B. Carlson, "Communication Systems: An Introduction to Signals, Noise, and Data," McGraw-Hill, 2013.

[3] J. G. Proakis, "Digital Communications," McGraw-Hill, 2001.

[4] R. E. Ziemer, "Elements of Modern Digital Communication," Prentice Hall, 1995.

[5] S. S. Haykin, "Communication Systems," John Wiley & Sons, 2001.

[6] M. S. Corrigan, "Amplitude Modulation," in "The Electrical Engineering Handbook," CRC Press, 2006.

[7] J. R. Barry, "Amplitude Modulation," in "The Communications Handbook," CRC Press, 2002.

[8] B. Carlson, "Communication Systems: An Introduction to Signals, Noise, and Data," McGraw-Hill, 2013.

[9] Janssen, T.; Weyn, M.; Berkvens, R. A primer on real-world RSS-based outdoor NB-IoT localization. In Proceedings of the 2020 International Conference on Localization and GNSS (ICL-GNSS), Tampere, Finland, 2–4 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6. [Google Scholar]

[10] Keerthika, M.; Shanmugapriya, D. Wireless sensor networks: Active and passive attacks-vulnerabilities and countermeasures. Glob. Transit. Proc. 2021, 2, 362–367. [Google Scholar] [CrossRef]

[11] Xiao, L.; Li, Y.; Han, G.; Liu, G.; Zhuang, W. PHY-layer spoofing detection with reinforcement learning

in wireless networks. IEEE Trans. Veh. Technol. 2016, 65, 10037–10047. [Google Scholar] [CrossRef]

[12] Basan, E.; Basan, A.; Nekrasov, A.; Fidge, C.; Sushkin, N.; Peskova, O. GPS-spoofing attack detection technology for UAVs based on Kullback–Leibler divergence. Drones 2021, 6, 8. [Google Scholar] [CrossRef]

[13] Zhang, P.; Nagarajan, S.G.; Nevat, I. Secure location of things (SLOT): Mitigating localization spoofing attacks in the Internet of Things. IEEE Internet Things J. 2017, 4, 2199–2206. [Google Scholar] [CrossRef]

[14] Jiang, Z.; Zhao, K.; Li, R.; Zhao, J.; Du, J. PHYAlert: Identity spoofing attack detection and prevention for a wireless edge network. J. Cloud Comput. 2020, 9, 5. [Google Scholar] [CrossRef]

[15] Khan, F.; Al-Atawi, A.A.; Alomari, A.; Alsirhani, A.; Alshahrani, M.M.; Khan, J.; Lee, Y. Development of a Model for Spoofing Attacks in Internet of Things. Mathematics 2022, 10, 3686. [Google Scholar] [CrossRef]

[16] Jullian, O.; Otero, B.; Stojilović, M.; Costa, J.J.; Verdú, J.; Pajuelo, M.A. Deep Learning Detection of GPS Spoofing. In Proceedings of the International Conference on Machine Learning, Optimization, and Data Science, Grasmere, UK, 4–8 October 2021; Springer: Cham, Switzerland, 2021; pp. 527–540. [Google Scholar]

