International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-11, September 2019

Development of Custom Interface Driver between USRP and SoC devices for SDR Applications

Boya Pradeep Kumar, Chandra Sekhar Paidimarry

Abstract:Global Navigation Satellite System (GNSS) is used to provide position based on satellite constellation. The GNSS Software defined radio (SDR) is a software approach which is more flexible than Hardware receivers. In GNSS SDR, most of the RF frontends are interfaced with a host-Personal Computer (PC). The host-PC based GNSS-SDR systems suffer from computational complexity and latency between IQ samples of GNSS signals. There are several RF front ends available to capture the real time signals. Among all these, USRP RF frontends are flexible for the prototype development. Earlier the host-PC was interfaced with the USRP device for the SDR implementation. In this work the, USRP N210 is interfaced with Zynq SoC device for GNSS SDR applications. This approach will introduce parallelism, which in turn leads to reduction in latency between IQ samples. A custom Ethernet driver to interface the USRP N210 with Zynq SoC device is proposed. The experimental results showcase that the proposed SoC based approach is suitable for GNSS-SDR prototype.

Keywords: GNSS, Software defined radio (SDR), RF, USRP, IQ, Zynq SOC device.

I. INTRODUCTION

Nowadays location based services and applications plays important role in the market. In the conventional Global Navigation Satellite System (GNSS) receivers, the baseband processing algorithms are performed in a dedicated chip. This Application Specific Integrated Circuit (ASIC) chip can't be reprogrammed once it is fabricated. In order to overcome this drawback, software based GNSS receivers are designed to receive and process the GNSS signals and provide user position. This software approach provides more flexibility to implement different GNSS algorithms without changing much hardware [1]. Over the past decade there is a tremendous revolution in the GNSS software receiver technology. Researchers are trying to developing a low cost GNSS software receiver by using FPGAs [2]. Generally in the GNSS software receivers the Analog to Digital Converter (ADC) is placed after the antenna to convert the GNSS signals from analog to digital format. The baseband processing algorithms are applied to the down converted digitized Intermediate Frequency (IF) Signal and processed in a programmable processor instead of in ASIC.

In the GNSS software receivers still hardware is required which is Radio Frequency (RF) frontend to capture the real time GNSS signals. The various RF frontends with different frequency bandwidths are available in the market for GNSS software receivers such as USRP, UmTrx, HackRF, BladeRF and ADI-FMComms, etc [3]. These RF frontends are connected to host PC or embedded system by using USB or Ethernet interfaces. Based on these interfacing devices, RF frontends and development environment, the GNSS software receivers can be grouped into two categories as depicted in the figure 1.

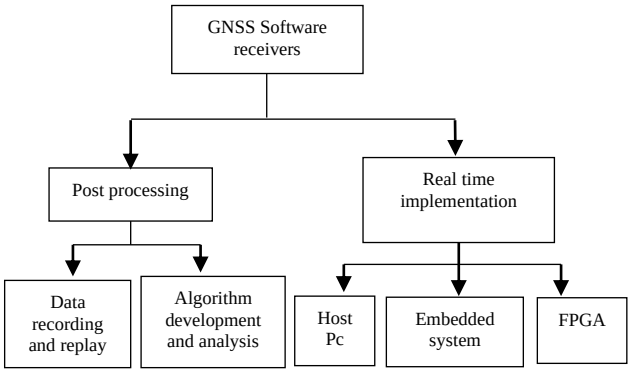

Figure 1. Different categories in GNSS software receivers

From the figure 1, the GNSS software implementation can be segregated into two categories which are post processing and real time implementation. Post processing consists of real time GNSS signal recording and algorithms development. This phase is carried out in the simulation environment. In real time implementation, the RF frontends are directly interfaced with either host PC or embedded platform or FPGA system. In this category the real time GNSS signals are processed under the real time environment to provide user position [4].

On a host PC or embedded system, the baseband processing algorithms can be developed in C++ programming language or Matlab environment [5]. In this approach, the host PC or embedded system have limited computational resources. The computational complexity to execute the baseband algorithms is also high [6].

Manuscript published on 30 September 2019.
*Correspondence Author(s)
Boya Pradeep Kumar*, Research Scholar, Department of Electronics and Communication Engineering, University College of Engineering, Osmania University, Hyderabad-500007, Telangana, India. pradeep.boge@gmail.com
Chandra Sekhar Paidimarry, Professor ,Department of Electronics and Communication Engineering, University College of Engineering, Osmania University, Hyderabad-500007, Telangana, India. sekharpaidimarry@gmail.com.

In an FPGA based system, programmable logic cells are used to execute these algorithms to reduce the computational complexity [7]. In this paper an interface between RF frontend and FPGA device is presented. The challenges involved in the interfacing between both devices are addressed. The USRP kit is used as RF frontend kit to capture real time GNSS signals and Zynq System on Chip (SoC) is used as FPGA device. The main contribution of this work is the development of custom interface driver between USRP and FPGA device.

Erick Schmidt et al [8] proposed and demonstrated a state-of-the-art GPS receiver based on LabView and C/C++ testbeds. In this work, Ettus USRP B200 with Global Positioning System disciplined oscillator (GPSDO) RF frontend kit is used to capture the GPS L1 signal. NUC 5i5RYK is used as host PC in order to process the GPS baseband algorithms. The authors demonstrated and characterized different acceleration techniques for real time SDR architecture for low cost platforms. Instead of using parallel code phase search acquisition algorithm, the joint-search FFT acquisition is used to find the visible GPS satellites.

Alexander M et al [9] disused the developments in the SDR technology. They conducted different case studies of communication algorithms in different SDR based hardware and software tools. The authors reported that, USRP kits are most affordable SDR platforms in order to prototype the communication algorithms.

Benjamin Drozdenko et al [10], introduced a method for Hardware (HW) – Software (SW) co-design for communication system on Zynq SoC based platforms. The authors used ADI-FMComms-3 as RF frontend for transceiver architecture to implement multiple protocols. The authors explore the issues in using HW-SW commercial tools such as Xilinx Vivado and Matlab-simulink. The proposed approach is suitable to implement different communication protocols for real time prototypes.

In most of literature, the authors are addressing the challenges in real time SDR implementation [3]. It is understood from the research articles and literature that the real time SDR architectures suffer from computational complexity and high HW cost [10]. The main contribution of proposed work is to establish the real time SDR platform by integrating the USRP N210 kit with Zynq SoC FPGA, to improve the computational speed. The proposed approach is cost effective when compared to the existing SDR systems.

The rest of the paper is as follows: Section II describes custom interface between USRP N210 and Zynq SoC FPGA device. This section also describes the implementation of custom interface driver. The results are discussed in section III. Finally, section IV concludes the paper.

II. CUSTOM INTERFACE USRP N210 AND ZYNQ SOC

The GPS receiver is primarily composed of three elements namely, antenna, controller and receiver hardware. Antenna is used to capture the GPS signal out of many electromagnetic signals that are available in the atmosphere. Once the GPS signals are captured, it is transferred to the receiver hardware to be processed by the baseband algorithms. The controller is loaded with GPS baseband processing algorithms. The commercial GPS receivers are not customizable and not flexible. The HW required for the setup is also very complex. For overcoming these disadvantages, new architectures and signal processing algorithms are needed. One such solution is obtained by using GNSS-SDR platforms. In GNSS-SDR, the entire baseband processing algorithms are performed in host PC or reconfigurable processors.

The GNSS SDR consists of RF front end and Processor. In most of the SDR architecture, a host PC is used to perform the GNSS algorithms. The host PC based GNSS SDR requires more computational time to provide the user position. To overcome this, an FPGA based GNSS SDR platform has been introduced. The main challenge involved in this work is to integrate USRP N210 RF frontend with Zynq SoC based FPGA device. There are no built-in drivers available to interface both these devices. The USRP N210 device uses Ethernet interface to communicate with other devices. In this proposed work a custom Ethernet interface has been developed in between USRP N210 and Zynq SoC.

A. USRP with host-PC based GNSS-SDR

The host-PC based GNSS-SDR consists of USRP RF front end and processor/ PC. The block diagram of USRP with host-PC based GNSS-SDR is depicted in figure 2. In this approach, initially the USRP device is used to capture the real time GNSS signals. A host PC is used to configure the USRP device for data storage and processing.

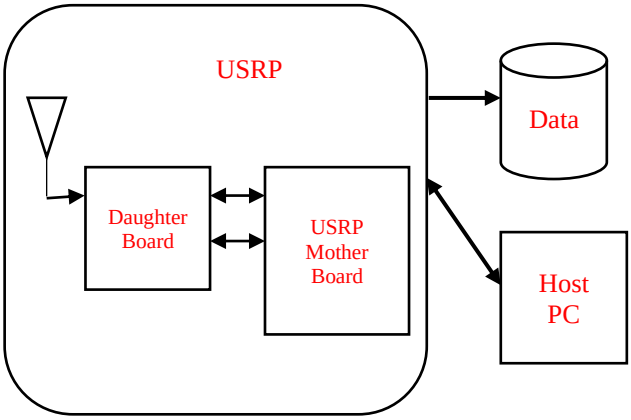

Figure 2. USRP with host-PC based GNSS-SDR

The GNU radio software (GRC tool) is used to implement various signal processing algorithms for SDR applications. The GNU radio is designed using either C++ or python on the host device and connected to the USRP device. This poses a challenge when computationally complex algorithms like FFT are used in such environment. These algorithms have to be executed on a parallel processing platform like FPGA where results can be obtained in parallel. Another issue is the latency caused in the IQ in between the USRP and GPU. This latency becomes negligible when the USRP front end is connected to the Zynq SoC through a Gigabit Ethernet port. In the literature, it is found that the researchers are using the native FPGA hardware without modifications in the USRP device. In the proposed approach, the FPGA is reconfigured with a new Ethernet driver to facilitate the interfacing process.

Retrieval Number: K19690981119/19©BEIESP
DOI: 10.35940/ijitee.K1969.0981119
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

B. Proposed model

In the proposed model, USRP N210 deice is interfacing with Zynq SoC device to reduce the latency which is depicted in figure 3. In this approach, the preliminary step is to load an Embedded Linux Operating System (OS) in Zynq SoC device to control the USRP device. In the second step the Linux OS is installed with the GNU radio tool. Finally the USRP device is connected with Zynq SoC by using Ethernet cable.

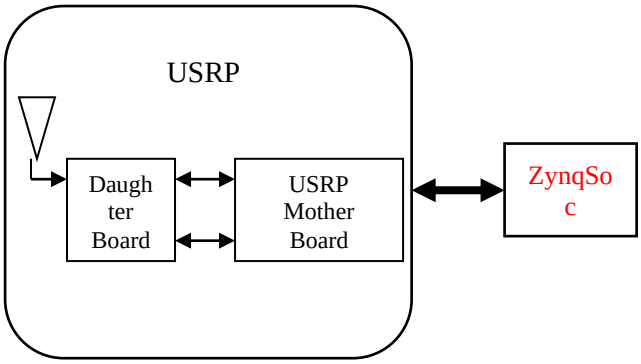

Figure 3. USRP with Zynq SoC based GNSS-SDR

The Zynq SoC with Embedded Linux OS consists "eth0" drivers which is not compatible for USRP device. This is a major bottle neck for interfacing both USRP and Zynq SoC devices. This challenge is overcome by using custom Ethernet driver as shown in figure 4.

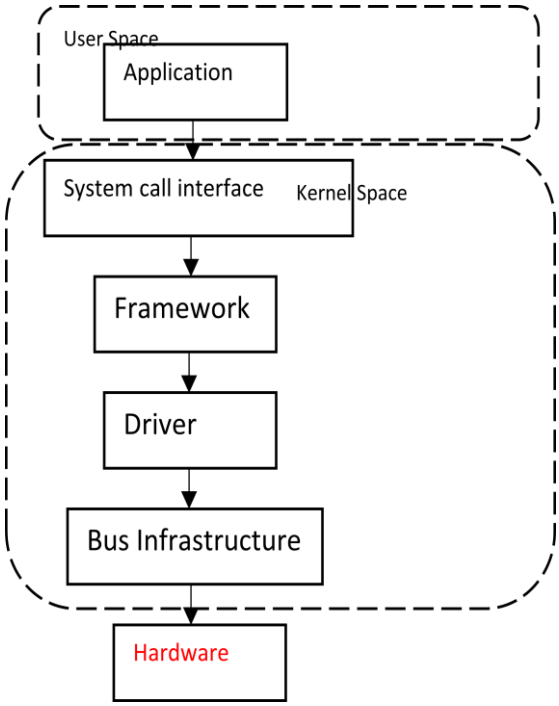

Figure 4. Custom Ethernet driver development flow

From the figure 4, the custom Ethernet driver development flow can be divided into two layers. The first layer is the application layer which is installed by the user. In this layer, the GNU radio tool is installed as application. The second layer is the kernel space which consists of system call interface, framework, driver and bus infrastructure. The custom Ethernet driver follows a framework which allows the driver to expose the Ethernet port features to user application. The driver block consists of modified scripts that communicates with the Hardware by using bus infrastructure. Once custom Ethernet driver is completed then it is loaded into Linux OS. The development of Linux OS with custom Ethernet driver is shown in figure 5.

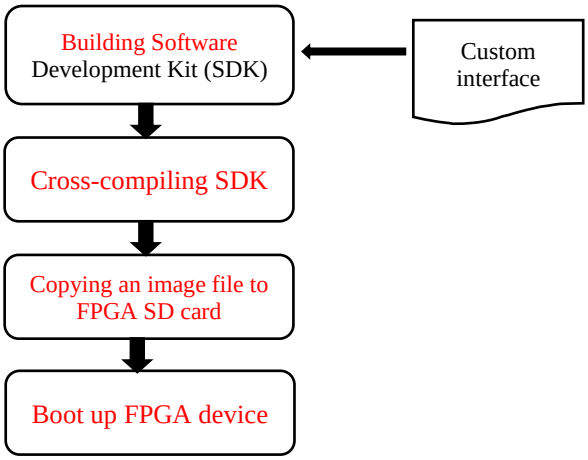

Figure 5. Development of Linux OS with custom Ethernet driver

The need for loading an OS on to the hardware is that the radio application needs an OS for installation into the hardware. There are a few applications which can run on the Zynq board without an OS. But these applications are too complex and the performance of the overall system degrades in such a process. Hence installing an OS into Zynq will make the applications to consume less power and more number of applications that can be used in such scenario also increases. Several steps are needed to port the Linux operating system on to a Zynq SoC. The tools are used in the process are dependent on the processor chosen. In this experiment, Zynq 7Z020 is chosen as the target device for the SDR platform. The simulation is carried out on the Xilinx Vivado Software Development Kit (SDK) platform. Initially in Vivado SDK platform, the custom interface driver is added through uBoot and First Stage Boot Loader (FSBL). After adding the custom interface driver the Boot.bin file has been generated by performing several steps in the SDK. In the next stage, the failures in the boot sequence have been verified by performing the cross compiling the SDK. Finally the boot files are copied in to SD card and then boot up the Zynq SoC device.

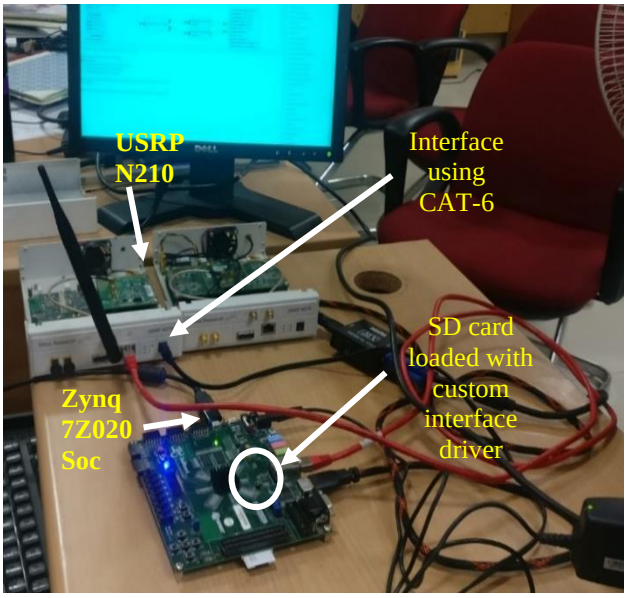

Figure 6. Photograph of interfacing USRP N210 with Zynq 7Z020 SoC

The hardware setup for interfacing the USRP N210 with Zynq SoC device is shown in figure 6. The experiment is carried out successfully and corresponding results with explanation are discussed in the next section.

III. RESULTS AND DISCUSSION

In this section the interfacing of USRP N210 RF front end device with Zynq SoC FPGA results has been presented. Initially USRP N210 device is configured with the IP 192.168.10.2. Then ethernet interface port "eth0" in the Zynq SoC device is configured with the IP address: 192.168.10.1. Both these devices should be connected in the same subnet mask which is 255.255.255.0. In the conventional approach, the Linux operating system which is loaded in the Zynq SoC FPGA is unable to find the USRP device due to unsupported Ethernet driver. The corresponding results is shown in figure 7.

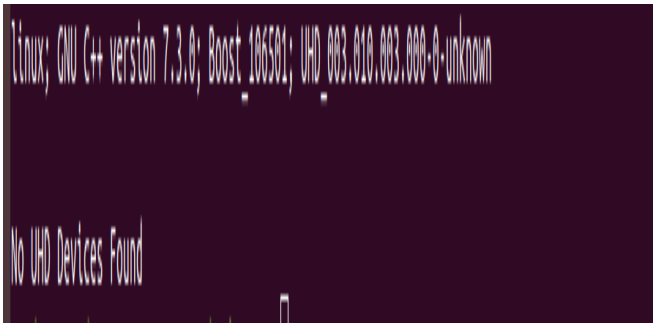

Figure 7. No USRP device found by using Convectional approach

In the proposed approach, the custom Ethernet driver is used to interface both USRP and Zynq SoC FPGA. In order to establish communication between the two devices, the custom Ethernet driver performs following tasks:

1. Identify the IP address of the USRP kit and obtain the MAC Address.
2. Enable the communication between USRP and FPGA

The Linux operating system with custom Ethernet driver is loaded in to Zynq SoC FPGA device. The USRP transfers the Ethernet packets continuously from USRP to Zynq SoC.

Figure 8 shows the results obtained by the proposed approach with custom Ethernet driver.

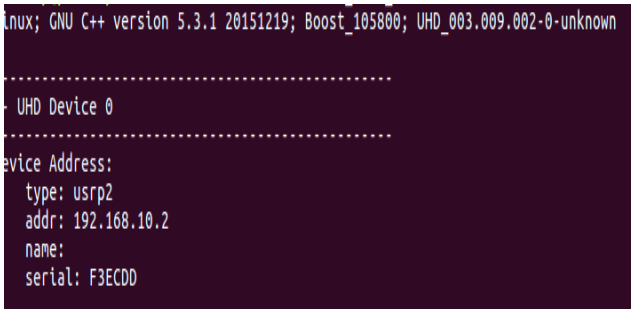

Figure 8. USRP device found by using proposed approach

The terminal provides the ID of the connected USRP device followed by details such as:

- Type of the device connected
- IP address
- Name of the device
- The serial number of the device.

This information is very crucial in the case where multiple USRP devices are connected to the same Zynq board. The target USRP device can be configured using the GNU radio software.

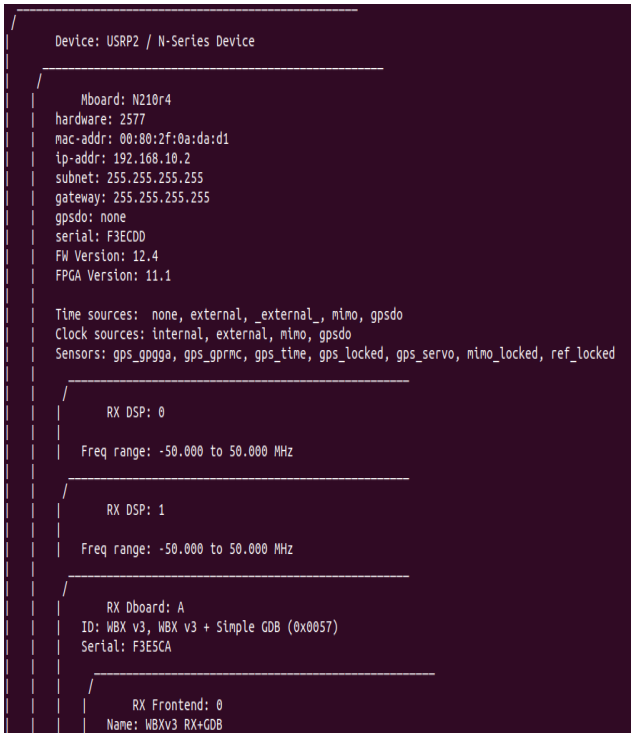

Figure 9. USRP Hardware specifications by performing *UHD_find_probe* command

Once the USRP device is selected, the following information is displayed on the terminal:

- Motherboard revision
- Hardware code
- MAC address
- IP address
- Subnet
- Gateway
- GPS do oscillator

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

- Device serial number
- Firmware version
- FPGA version
- Clock and time sources
- Receiver specifications
- Daughter board specifications
- Connected Antennas

Figure 9 shows the hardware configuration used in the experiment. The mother revision is N210r4 and the hardware code is 2577. The MAC address and IP address are 00:80:2f:0a:da:d1 and 192.1198.10.2 respectively. The Subnet mask is 255.255.255.255 and gateway is 255.255.255.255. The serial number is F3EC2D and the Firmware version is 12.4. The FPGA version 11.1 specifying that the hardware is FPGA Spartan 3. Receiver specifications presents the number of Rx and Tx ports available in the USRP device. The figure displayed them as Rx DSP 0 and Rx DSP 1. The frequency range of the devices are also specified ie. -50 to 50 MHz. The Daughter board connected to the USRP is WBX having a frequency from 68 MHz to 2.2 GHz. The antennas connected to the devices are also listed. The experimental results show that the proposed approach helps to establish the connection between Zynq SoC and the USRP. This driver setup is useful for SDR applications.

IV. CONCLUSION

This paper presents the development of Custom Interface Ethernet driver for establishing the communication between USRP N210 and Zynq SoC. Building an application in SDK involves running the library generator tool which generates the Linux OS.In the Linux operating system each peripheral is communicated with the help of kernels. The default *'eth0'* driver is replaced with the custom Ethernet Driver which enables the communication interface. This driver enables the user to prototype the GNSS SDR for real-time development.

ACKNOWLEDGEMENT

We wish to thank Council of Scientific & Industrial Research (CSIR), India, for providing financial support for carrying out this work. We are thank University Grants Commission (UGC) India for providing the experimental setup under UGC Major Research Project (File No: F.41-609/2012 (SR) Dated 16/07/12).

REFERENCES

1. Borre. K, Akos. D.M, Bertelsen. N, Rinder. P, and Jensen. S.H, "A Software-Defined GPS and Galileo Receiver: A Single-Frequency Approach", Basel, Switzerland, Birkhauser publishing, 2007.
2. Surabhi Guruprasad, Sunil Bisnath, Regina Lee and Janusz Kozinski, "Design and implementation of a low-cost SoC-based software GNSS receiver" IEEE Transactions on Aerospace and Electronics System-Volume: 31, Issue: 4, April 2016.
3. Alexander M. Wyglinski, Don P. Orofino, Matthew N. Ettus and Thomas W. Rondeau, "Revolutionizing software defined radio: case studies in hardware, software and education" IEEE Communications Magazine , Volume: 54 , Issue: 1, January 2016.
4. Jong-Hoon Won, Thomos Pany and Gunter W.Hein, "GNSS Software Defined Radio Real Receiver or Just a Tool for Experts?" Inside GNSS, July/Aug 2006.
5. Kwi Woo Park, Yun Sub Choi, Min Joon Lee, Sang Jeong Lee and Chansik Park "Implementation and Performance Analysis of Multi-GNSS Signal Collection System using Single USRP " Journal of Positioning, Navigation, and Timing, Volume 5, Issue 3 - Sep 2016.
6. Mamatha Ramapura Maheshwarappa, Mark D. J. Bowyer and Christopher P. Bridges "Improvements in CPU & FPGA Performance for Small Satellite SDR Applications" IEEE Transactions on Aerospace and Electronic Systems, Volume: 53 , Issue: 1 , Feb. 2017.
7. Carlos Ribeiro and Atı´lio Gameiro, "A software-defined radio FPGA implementation of OFDM-based PHY transceiver for 5G" Analog Integrated Circuits and Signal Processing Journal (Springer), Volume 91, Issue 2, pp 343–351, May 2017.
8. Erick Schmidt, David Akopian and Daniel J. Pack "Development of a Real-Time Software-Defined GPS Receiver in a LabVIEW-Based Instrumentation Environment" IEEE Transactions on Instrumentation and Measurement, Volume: 67 , Issue: 9 , Sept. 2018.
9. Alexander M. Wyglinski, Don P. Orofino, Matthew N. Ettus and Thomas W. Rondeau, "Revolutionizing software defined radio: case studies in hardware, software and education" IEEE Communications Magazine, Volume: 54, Issue: 1, January 2016.
10. Benjamin Drozdenko , Matthew Zimmermann, Tuan Dao, Kaushik Chowdhury and Miriam Leeser, "Hardware-Software Codesign of Wireless Transceivers on Zynq Heterogeneous Systems" IEEE Transactions on Emerging Topics in Computing, Volume: 6 , Issue: 4 , Oct.-Dec. 1 2018.
11. K.Praveen Kumar, "Design of 3D EBG for L band Applications" IEEE International conference on communication technology ICCT-April-2015. Noor Ul Islam University Tamilnadu.
12. K.Praveen Kumar, "Effect of 2DEBG structure on Monopole Antenna Radiation and Analysis of It's characteristics" IEEE International conference on communication technology ICCT-April-2015. Noor Ul Islam University Tamilnadu.
13. K.Praveen Kumar, Dr Habibulla Khan " The surface properties of TMMD-HIS material; a measurement" IEEE International conferance on electrical, electronics, signals, communication & optimization EESCO - January 2015.
14. B. Venkateswar Rao, Praveen Kumar Kancherla, Sunita Panda "Multiband slotted Elliptical printed Antenna Design and Analysis" Journal Of Mechanics Of Continua And Mathematical Sciences (JMCMS), Vol.-14, No.-4, July-August (2019) pp 378-386.
15. Kumaraswami Gajula, Amulya Boyina, K. Praveen Kumar "Active Quad band Antenna Design for Wireless Medical and Satellite Communication Applications" Journal Of Mechanics Of Continua And Mathematical Sciences (JMCMS), Vol.-14, No.-4, July-August (2019) pp 239-252.
16. Amulya Boyina, K. Praveen Kumar "Active Coplanar Wave guide Fed Switchable Multimode Antenna Design and Analysis" Journal Of Mechanics Of Continua And Mathematical Sciences (JMCMS), Vol.-14, No.-4, July-August (2019) pp 188-196.