FPGA-Based Software-Defined Radio and Its Real-Time Implementation Using NI-USRP

Nikhil Marriwala , Om. Prakash. Sahu and
Anil Vohra

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/66272

Abstract

In this chapter, we propose a novel design of scalable and real-time data acquisition software architecture for software-defined radio (SDR) using universal software radio peripheral (USRP). The software has been designed and tested in multi-thread model, using LabVIEW, which guarantees real-time performance and efficiency. With the help of this design, we have been able to improve the stability of the system besides providing a reconfigurable and flexible architecture. Wireless transfer of sensitive data using communication is not a very safe option. In this chapter, we aim to provide a safe and private wireless transmission between two terminals using the SDR approach and verifying the results in real-world environment with the use of USRP. The novel design being presented here can be used to transfer (random data, text or an image) encoded with different forward error correction (FEC) codes, which is then verified at the receiving terminal and then decoded accordingly to produce the desired result.

Keywords: software-defined radio, universal software radio peripheral, forward error correction codes, biterrorrate (BER), signal-to-noise ratio (SNR), LabVIEW

1. Introduction to SDR

Software-defined radio (SDR) systems are those which can adapt to the future-proof solution and they cover both existing and emerging standards. An SDR has to possess elements of reconfigurability, intelligence and software programmable hardware [1]. As the functionality is defined in software, a new technology can be easily implemented in a software radio by means of a software upgrade. Channel equalization is an important subsystem in the software defined radio's receiver. For many years, modulation techniques have been extensively used for various wireless applications, but the modern communication system requires data transmitted at a higher rate and larger bandwidth [2].

This chapter discusses an SDR system built using LabVIEW for a generic transceiver. SDR provides an alternative to systems such as the third-generation (3G) and the fourth-generation (4G) systems. There are two frequency bands where the software-defined radio might operate in the near future, that is, 54–862 MHz [very high-frequency (VHF) and ultra-high frequency (UHF) TV bands] and 3–10 GHz [ultra-wideband (UWB) radios]. A software-defined radio comprises a programmable communication system where functional changes can be made by merely updating the software. SDRs can be reconfigured and can talk and listen to multiple channels at the same time. Normally, high-performance digital signal processors (DSPs) are used to serve as the baseband processor. SDR systems can be used in ubiquitous network environments because of its flexibility and programmability [3]. The use of digital signals reduces hardware, noise and interference problems as compared to the analogue signal in transmission, which is one of the main advantages of digital transmission.

In this chapter, the software simulator of the SDR transceiver has been designed using LabVIEW and has been tested in real time using the universal software radio peripheral (USRP). Digital modulation schemes namely the frequency shift keying (FSK), phase shift keying (PSK) and quadrature amplitude modulation (QAM) are chosen to be the modulation schemes of the designed software-defined radio system due to its easy implementation and widespread usage in wireless communication equipment. Digital modulation techniques are considered to be very common technology for transmission and reception in current and future wireless communications, especially in the VHF and UHF frequency bands giving excellent bit error rate (BER) versus signal-to-noise ratio (SNR) ratio with high data rates. The role of modulation techniques in an SDR is very vital given that the modulation techniques describe the central part of any wireless technology. They can be reconfigured and can talk and listen to multiple channels at the same time. The role of modulation techniques in an SDR is very crucial since modulation techniques define the core part of any wireless technology. SDR's inherent flexibility must, however, be planned for in advance via hardware and software considerations, ultimately resulting in increased code portability, improved communications system life cycles and reduced costs. The main interest in any communication group is the sure sending of signals of info from a transmitter to a receiver [4]. The signals are transmitted via a guide who corrupts the signal. It is needful that the distorting effects of the channel and noise are minimized and that the information transmitted through the channel at any given time is maximized [5]. The channel is subject to various types of dissonance, twisting and interference. Also, some communication systems have limitations on transmitter power. All of this may lead to various types of errors. Consequently, we may need some form of error control encoding in order to recover the information reliably.

1.1. Related technologies of SDR

In view of the fact that the field-programmable gate array (FPGA)'s prime function is to offer signal filtering and rate conversion from the analogue-to-digital converter (ADC) and to the system's digital-to analogue converter (DAC), its firmware can be customized to meet the particular needs of the user or just downloaded without modification, as a preconfigured file. Some of the USRP family support multi-radio cooperation using multiple-input, multiple output (MIMO) techniques. This is enabled by installing a MIMO interconnecting cable between two USRP devices [6].

To ensure reliable communication, forward error-correcting (FEC) codes are the main part of a communication system. FEC is a technique in which we add redundant bits to the transmitted data to help the receiver correct errors. There are two types of FEC codes: the convolutional codes and block codes. When we use block codes, they are defined by n and k, where n describes the total number of coded bits and k gives the number of input bits. In convolutional codes, the coding is applied to the entire data stream as one code word. In the year 1948, Shannon showed that arbitrarily reliable communication is only possible till the signal transmission rate does not exceed a certain limit which was termed as channel capacity [7]. After this different algebraic codes such as Golay codes, Bose-Chaud huri-Hocquenghem (BCH) codes and Reed-Solomon (RS) codes were created and used for error correction. The next series of codes originally referred as recurrent codes or convolutional codes were given which helped further to improve the error control coding [8]. The convolutional codes have efficient encoding and decoding algorithms and high performance over additive white Gaussian noise (AWGN) channels. Later on, concatenated-coding schemes were also given. Also, some weak points were there of convolutional codes during bursty transmissions which were later on reduced using Reed-Solomon codes by serially concatenating a convolutional code with an RS code. Development of turbo codes is the most recent discovery in the coding theory. Turbo codes show performance of near to Shannon limit with iterative decoding algorithms. Many iterative decoding algorithms came into existence such as Gallagher's low-parity density check (LDPC) codes [9]. Though these turbo codes exhibit excellent bit error performance but there are some problems associated with them such as these codes generate a certain number of low-weight code words which results in exhibition of an error 'floor' in the BER curve at high SNR. Also, the complexity of the soft-input, soft-output (SISO) decoder is such that low-cost decoders are unavailable for many commercial applications [10]. For these reasons, many applications still deploy RS codes because of its efficient decoder implementation and excellent error correction capabilities.

1.2. The benefits of multi-standard terminals

A multi-standard terminal (MST) is a subscriber unit that is capable of operation with a variety of different mobile radio standards. Although it is not strictly necessary for such a terminal to be implemented using software-defined radio techniques, it is likely that this approach is the most economic in many cases [11].

1.2.1. Economies of scale

Even if terminal adaption over the air or via third-party software was not possible or was not permitted by, for example, regulatory bodies, the production benefits of a software-defined radio approach could well justify its existence. The wide range of new and existing standards in the cellular and mobile marketplace has resulted in the adoption of a diverse range of subscriber terminal (and base-station) architectures for the different systems deployed around the globe [12].

1.2.2. Global roaming

The present proliferation of mobile standards and the gradual migration to third-generation systems means that a large number of different network technologies will exist globally for some time to come. Indeed, even in the case of 3G systems, where a concerted effort was made by international standards bodies to ensure that a single global standard was produced, there are still significant regional differences, in particular between the US and European offerings (and also, potentially, China). With this background, it is clearly desirable to produce a terminal which is capable of operation on both legacy systems and the various competing 3G standards. Indeed, it could be argued that this is the only way in which 3G systems will be accepted by users, since the huge cost of a full-coverage network roll-out will discourage many operators from providing the same levels of coverage (at least initially) as their existing 2G systems enjoy. A user is unlikely to trade in his or her 2G terminal for one with perhaps better services, but significantly poorer basic voice coverage [13]. This is indeed what is happening in virtually all current 3G deployments. Software-defined radio architecture represents a very attractive solution to this problem.

1.2.3. Upgrading the service

A powerful benefit of a software-defined radio terminal, from the perspective of the network operator, is the ability to download new services to the terminal after it has been purchased and is operational on the network. At present, significant service upgrades require the purchase of a new terminal, with the required software built-in and this clearly discourages the adoption of these new services for a period. The launch of general packet radio service (GPRS) data services on the GSM network is a good example of this. With SDR handset architecture, services could be downloaded overnight, when the network is quiet, or from a website in the same manner as personal computer (PC) software upgrades are distributed [12]. There are clearly a number of logistical issues with this benefit (e.g., what to do about phones which are turned off at the time of the upgrade or what happens if a particular phone crashes with the new software, perhaps just prior to requiring the phone for an emergency call—software which the phone user may not have wanted and so forth). Many of these problems have been solved by the PC industry and hence it is likely that this benefit will be realized in some manner with software-defined radios.

1.2.4. Adaptive modulation and coding

The ability to adapt key transmission parameters to the prevailing channel or traffic conditions is a further key benefit of a software-defined radio. It is possible, for example, to reduce the complexity of the modulation format, such as from 16-QAM (quadrature amplitude modulation) to quadrature phase-shift keying (QPSK) when channel conditions become poor, thereby improving noise immunity and decoding margin. It is also possible to adapt the channel-coding scheme to better cope with particular types of interference, rather than Gaussian noise, when moving from, say, a rural cell to an urban one [14]. Many parameters may be adapted dynamically, for example, burst structure, modulation type, data rate, channel and source coding, multiple-access schemes and so forth.

1.3. Operational requirements: various operational requirements for SDR are as stated below:

1.3.1. Key requirements

The operational characteristics of an ideal multi-standard terminal include the following operations.

1.3.1.1. Software-definable operation

As outlined earlier, the key to many of the advantages of a multi-standard terminal lies in its ability to be reconfigured either during manufacture, prior to purchase, following purchase (e.g., after-market software), in operation (e.g., adaptation of coding or modulation), or preferably all four. This impacts primarily upon the digital and baseband sections of the terminal and will require the use of reprogrammable hardware as well as programmable digital signal processors in a power and cost-effective implementation.

1.3.1.2. Multi-band operation

The ability to process signals corresponding to a wide range of frequency bands and channel bandwidths is a critical feature of an MST. This will impact heavily on the radio frequency (RF) segments of the terminal and it is this area which is arguably the main technology limitation on software-defined radio implementation at present [15] (although processor power consumption and cost are still both major issues for SDR).

1.3.1.3. Multi-mode operation

Many multi-mode software-defined radios already exist, although they are often not promoted as such (since the other features/benefits of software-defined radio techniques are not exploited). The ability to change mode and, consequently, modulation, coding, burst structure, compression algorithms and signalling protocols is clearly an essential feature of an MST.

1.4. Digital aspects of a software-defined radio

1.4.1. Digital hardware

There exists a range of solutions to the digital-processing problem for a software-defined radio, each with its own characteristics and application areas. The digital-processing area is, in many respects, as challenging as the analogue processing described in detail in this book and the intention of this section is merely to highlight the options and their main characteristics. The two biggest issues at present are the power consumption and cost of the various options. In a base-station application, these are less of an issue (but are still a significant challenge); however, they are perhaps the main inhibitor to the widespread use of software-defined radio in handsets and other portable devices:

1. The use of reconfigurability as a method of providing upgrading, improvement or backwards compatibility (i.e., a smooth transition from a legacy system) is, however, a strong argument for flexible processing and SDR concepts. It is in this context that the processing options outlined in the following will be discussed.

Cost is also a multi-faceted issue. Most designs judge cost based almost exclusively on the cost of the target device used for the code (be it a processor or an FPGA). In the case of a very high-volume application (e.g., a handset), this might be a reasonable approach, although even here it could be somewhat short-sighted. In the case of a base-station design, however, there are many other considerations that will determine the overall cost of a design (particularly if lifetime cost is considered and not just purchase cost). As a summary, the factors that influence the cost of the digital elements of an SDR BTS include

a. Direct cost of the processing device itself.

b. Costs involved in the associated ancillary and interfacing devices (e.g., memory, clock circuitry and so forth).

c. Non-recurring expense (NRE). This is most obviously associated with application-specific integrated circuit (ASIC) or application-specific signal processor (ASSP) designs and includes mask-set costs, fabrication and so forth. These costs are rising dramatically as feature sizes reduce and are therefore making the break-even volume (compared to, say, FPGAs) much higher as time progresses.

d. Tools/training investment. Changing from one digital technology to another (e.g., from DSPs to FPGAs) may well involve a significant change of design personnel, or at the very least a degree of retraining. This will have an associated cost and also an opportunity cost as the time to market will be increased (see the following). Even changing from one manufacturer's processors to another may involve a loss of productivity while the development team familiarizes itself with a new feature set and the new tricks required to get the best out of a particular device.

e. Cooling. The cost of cooling can undergo step changes as the form of cooling required changes. The most obvious example is in going from convection cooling to forced-air cooling, with the cost of the fans now needing to be added to the bill of materials. Additional power consumption will also add to the cost of the power supply, although with modern switched-mode designs, this is usually small. It is, however, a much bigger issue in handset designs due to the increased requirements it places upon the battery and the user-acceptance issues of large batteries or reduced talk times.

f. Development time/resource. This is becoming an increasingly important aspect of cost, as product life cycles, even of base-station designs, reduce as each new design appears. The volume of units sold of a particular design is then lower and the cost of producing that design becomes an ever-larger proportion of its selling price. Techniques or architectures, which allow these designs to be generated quickly, or significant portions of designs to be reused between evolutionary models in a range (as well as across models in a given range), are clearly attractive, even if the devices upon which they are based are not the lowest-cost components available.

g. Flexibility. This is a benefit in terms of time to market for new products and hence a benefit in terms of opportunity cost. If full flexibility could be provided for the same cost as a fixed solution (e.g., a single-application ASIC), then it would be a simple decision to adopt a flexible approach. This is almost never true and hence a full business case must be developed for flexibility, in a given marketplace and each opportunity judged on its merits.

1.4.1.1. Digital signal processors

DSPs were arguably the original enabling technology for software-defined radio (other than perhaps in military circles where cost is less important). They have the advantage of complete flexibility, wide applicability and a wide availability of skilled practitioners in their software. They are also high-volume devices and hence the benefits of economies of scale may be realized across a large number of applications in a wide range of industries (not just wireless communications). This, in general, makes up for their lack of optimization for a given specific project or niche application area and allows them to be a realistic option for early prototyping and initial production volumes of a new design, as well as for the final volume product, in some cases.

They are best suited to the less computationally intensive forms of signal processing, rather than very high-speed front-end applications. They are often utilized for involved, off-line processing of data which has been acquired and undergone initial processing/storage by a different type of device (e.g., an FPGA or an ASIC) [16]. They are, however, well supported and also tend to come in backwards-compatible families, which allow development to take place on a state-of-the-art (SOTA) device, with the final application device being lower cost. This generally occurs for two reasons:

1. The SOTA device being used in development will not be SOTA by the end of the development cycle and hence will generally have reduced in cost. The volume of usage of the device will also have increased, which will also help to reduce its cost.
2. Developers tend to pick a device for their development systems which is definitely large enough to meet the requirement in question. It is often the case that once development is nearing completion, the design will have been optimized such that it may be executed in a lesser member of the same device family. This will have an associated cost benefit. In larger systems, it may be the number of devices that can be reduced; however, this will still result in a lower overall cost.

1.4.1.2. Field-programmable gate arrays

FPGAs have undergone a revolution in recent years, both in performance and in cost. From humble beginnings as simple, flexible glue logic in complex digital designs, they are now a credible processing platform in their own right and able to rival ASIC solutions in many areas (and act as a low-cost prototyping mechanism for ASIC designs). They have also undergone a revolution in volume pricing, which means that they are no longer consigned to the prototype and initial volume parts of the product life cycle, but can now be used throughout volume production, in some applications [17]. It is also possible to convert from an FPGA to a quasi-ASIC, with a highdegree of confidence of success and a relatively low NRE (and hence break-even volume). FPGAs are therefore challenging and displacing ASICs in traditional ASIC application areas.

Furthermore, they provide much more flexibility than can be cost-effectively built into an ASIC, thereby fitting with the requirements of SDR very well. In common with DSPs, they also tend to come in families, thereby, again, allowing an initial design to take place on a large (potentially over specified) device with the final device being chosen to just fit the processing requirement. It is also possible (but not necessarily economic) to add IP processor cores into an FPGA (or an FPGA-derived ASIC). This makes possible a single-chip solution in some applications and this may be important for size or reliability reasons (with the improved reliability coming from the reduction in devices and soldered joints).

1.4.1.3. ASICs

The main issue with utilizing ASICs (or, more correctly, application-specific signal processors) within an SDR system lies in their lack of flexibility (or conversely the cost of adding flexibility). There are many methods by which flexibility may be introduced within an ASSP and these include the following:

a. Provision of multiple toolbox functions with flexible input parameters. An example would be a QAM modulator that had an input variable to configure it from 16 to 256 QAM, for example [18].

b. Provision of hardware for all current modulation formats, coding schemes and so forth in a single (large!) ASSP, with the ability to select between the different paths. This is not strictly flexible in the generic sense; however, it is flexible in its range of functionality—the user will not care how he is provided with service over a range of standards, just that he obtains service at a low cost. The major disadvantage with this option is that it is not really future-proof, unless the system designer has an extraordinary insight into the future trend in mobile communications (and can convince his or her management that he or she is right).

c. A combination of one or both of the above with some programmable DSP functionality (e.g., using an embedded DSP core). The key here is in providing enough DSP power to be useful and provide a degree of future-proofing, without designing essentially a DSP device—it would almost certainly be lower cost to buy an off-the-shelf DSP device from a volume vendor.

1.5. About NI-USRP

The NI USRP-2922 can be used for various communication applications, including

a. WiFi

b. WiMax

c. S-band transceivers

d. 2.4-GHz ISM band transceivers.

The NI USRP-2922 as shown in **Figure 1** has the following basic characteristics:

i. Frequency range of 400 MHz to 4.4 GHz

ii. Tunable RF transceiver

iii. Transmits and receives bandwidth up to 40 MHz

iv. High-speed ADC and DAC for streaming baseband I and Q signals to a host PC over gigabit Ethernet.

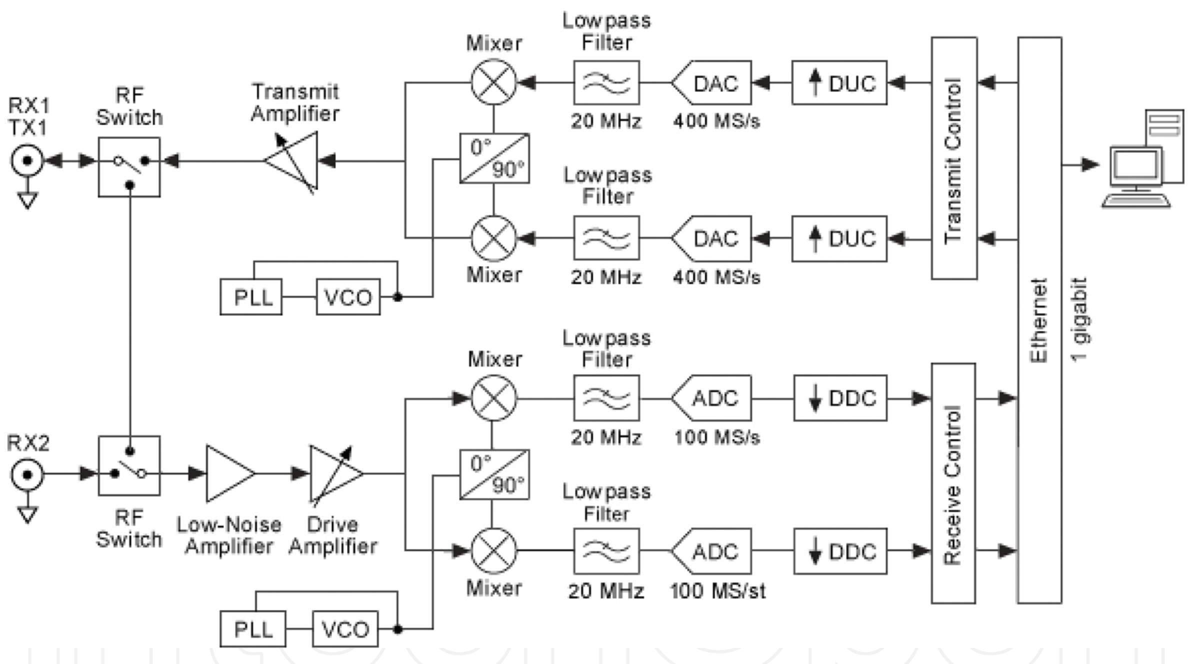


Figure 1. Block diagram of NI USRP-2922.

The RF switch allows transmit and receive operations to occur on the same shared antenna. On the NI USRP-2922, one antenna is designated receive-only.The NI USRP is also capable of receiving the signal, where the received signal is mixed down from RF using a direct-conversion receiver to baseband I/Q components. The digitized I/Q data follows parallel paths through a digital down-conversion (DDC) method that mixes, filters and decimates the input signal to a user-specified rate. Gigabit Ethernet connection is used to pass the down-converted samples to the host computer.

1.5.1. Receive trail

i. The received signal is amplified using the low-noise amplifier and drive amplifier amplify.

ii. The voltage-controlled oscillator (VCO) is controlled by the phase-locked loop (PLL) so that the device clocks and local oscillator (LO) can be frequency-locked to a reference signal.

iii. The mixer down-converts the signals to the baseband in-phase (I) and quadrature-phase (Q) components.

iv. To reduce the noise and high-frequency components in the signal, low-pass filter is used.

v. The analogue-to-digital converter digitizes the I and Q data.

vi. The digital down-converter (DDC) mixes, filters and decimates the signal to a user-specified rate.

vii. A standard gigabit Ethernet connection is used to pass the down-converted samples.

1.5.2. Transmit trail

i. The host computer synthesizes baseband I/Q signals and transmits the signals to the device over a standard gigabit Ethernet connection.

ii. The digital up converter (DUC) mixes, filters and interpolates the signal to 400 MS/s.

iii. The digital-to-analogue converter converts the signal to analogue.

iv. The low-pass filter reduces noise and high-frequency components in the signal.

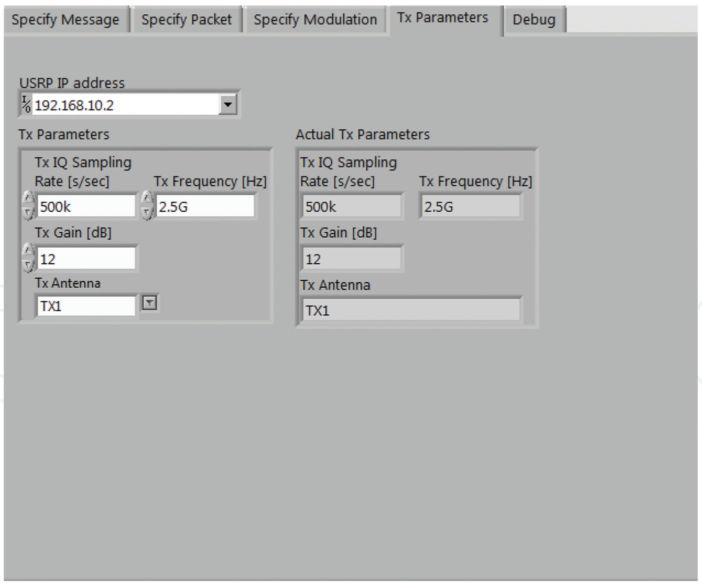
v. The mixer is used to up-convert the received signals to a user-specified RF frequency.

vi. The PLL controls the VCO so that the device clocks and LO can be frequency-locked to a reference signal.

vii. The transmit amplifier amplifies the signal, which is then transmitted through the antenna.

1.6. Design of a generic transceiver for FPGA-based SDR

In this section, the building blocks of a generic transceiver for FPGA-based SDR modem system built in LabVIEW have been explained. The designed system consists of two parts: the transmitter section and the receiver section. The transmitter section consists of four modules which are a message source module, pulse-shaped filter module, QAM modulator module and Gaussian noise module. The receiver has been designed using an adaptive filter module, QAM demodulator module, sync and tracking module. A brief description of each block follows. The front panel of generic transceiver for SDR modem system is shown in **Figure 2**.



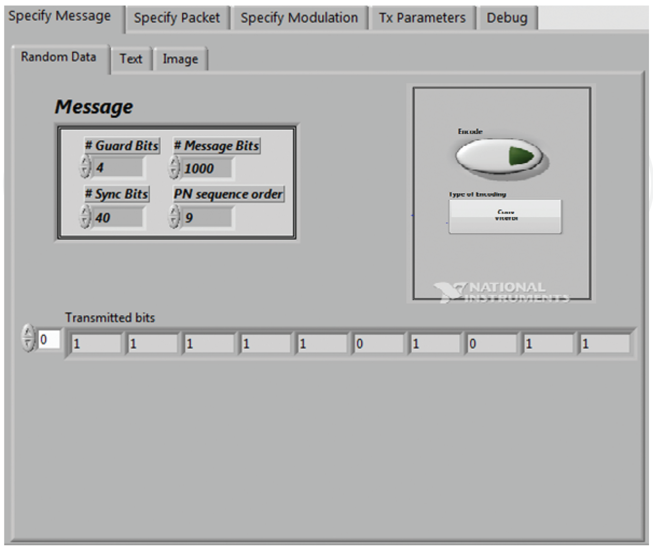
USRP IP address: 192.168.10.2

Tx Parameters
- Tx IQ Sampling Rate [s/sec]: 500k
- Tx Frequency [Hz]: 2.5G
- Tx Gain [dB]: 12
- Tx Antenna: TX1

Actual Tx Parameters
- Tx IQ Sampling Rate [s/sec]: 500k
- Tx Frequency [Hz]: 2.5G
- Tx Gain [dB]: 12
- Tx Antenna: TX1

Figure 2. Front panel of transmitter, Tx parameter.

In **Figure 2**, under Tx parameter is used to select the parameters such as IQ sampling rate, Tx frequency, Tx gain and so on of the transmitter. In the **USRP, IP address one s**elects the IP address of the transmitter device such as 192.168.10.2. In the **Tx IQ sampling rate [s/s] field, we must s**et the sampling rate to 500 k for text, random data and 2M for image transmission. **Tx frequency [Hz] is used to s**et the frequency to 2.5 G (the user can use any frequency from 400 MHz to 4.4 GHz for USRP2922). In the **Tx gain [dB]** field, we enter Tx gain 12 and we can take upto 31. Now, one need to select the Tx antenna as Tx1. In the specify message window in the transmitter front panel, we can select the type of the file to be transmitted;it consists of three files: random data, text and image as shown in **Figure 3**.

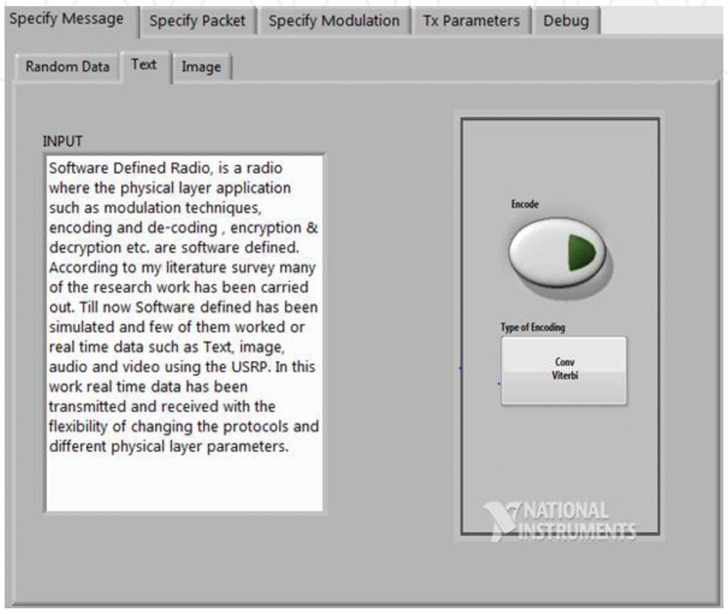


Message
- *# Guard Bits*: 4
- *# Message Bits*: 1000
- *# Sync Bits*: 40
- *PN sequence order*: 9

Transmitted bits: 0 | 1 1 1 1 1 0 1 0 1 1

Figure 3. Specify message/random data window.

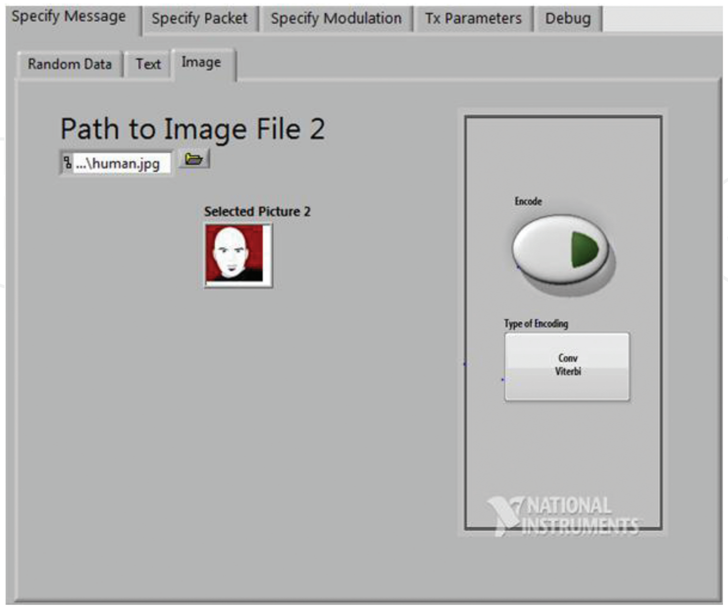
Random data can be transmitted by selecting the random data field. The message field is set to set the data to be transmitted. Transmitted bits indicate the bits which have to be transmitted. Encode button is meant to encode the transmitted data. Type of encoding is used to select the type of encoding required either convolution viterbi or turbo coding. Similar to the random data, one can select text or image window in order to transmit text or image, respectively. **Figures 3** and **4** show text and image window, respectively.



INPUT

Software Defined Radio, is a radio where the physical layer application such as modulation techniques, encoding and de-coding , encryption & decryption etc. are software defined. According to my literature survey many of the research work has been carried out. Till now Software defined has been simulated and few of them worked or real time data such as Text, image, audio and video using the USRP. In this work real time data has been transmitted and received with the flexibility of changing the protocols and different physical layer parameters.

Figure 4. Specify message/text window.



Path to Image File 2

...\human.jpg

Selected Picture 2

Figure 5. Specify message/image window.

We can write the text to be transmitted in the input window as shown in **Figure 4**. Encode button is used to provide encoding scheme. One needs to press the encode button to encode the transmitted data. The type of encoding field is used to select the type of encoding required either convolution viterbi or turbo coding as shown in **Figure 4**. Similarly, one can transmit the image as shown in **Figure 5**. We need to select the path or browse the path of the image and the selected picture is indicated in the front panel.

Pulse-shaping filter parameters field is used to set the system filter parameters as shown in **Figure 5**:

a. Tx filter: Used to select the type of filter required raised cosine filter or root raised cosine or none.

b. Alpha: Set the alpha to 0.50.

c. Filter length: Set filter length to 6.

Samples per symbol: Set samples per symbol to 8.

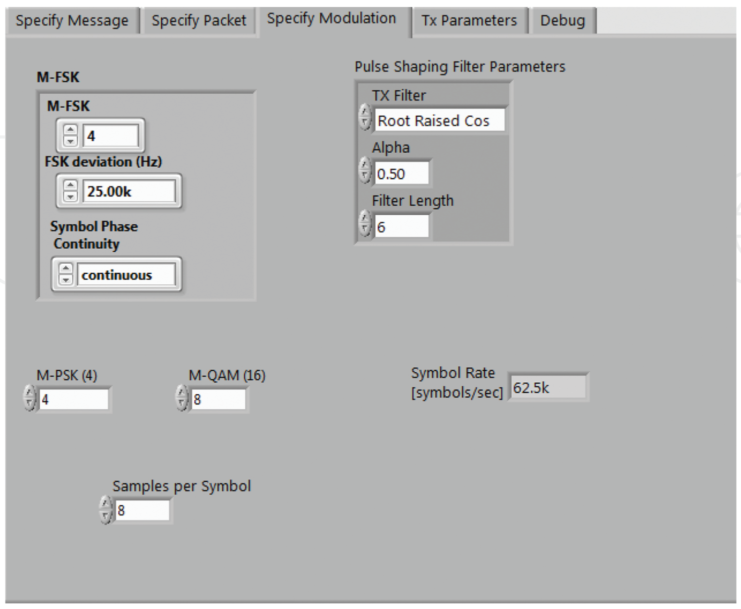
Symbol rate [symbols/s]: It indicates the symbol rate **M-FSK:** Set the FSK parameters as shown in **Figure 5**.

a. M-FSK: Set M-FSK to 4 (can be changed).

b. FSK deviation [Hz]: Set FSK deviation to 25.00 k.

c. Symbol phase continuity: Select symbol phase continuity to continuous.

M-PSK: Set M-PSK to 4. **M-QAM:** Set M-QAM to 8. Now, select the specific packets window from the transmitter front panel to set the message bits as shown in **Figure 6**.



M-FSK

M-FSK: 4

FSK deviation (Hz): 25.00k

Symbol Phase Continuity: continuous

Pulse Shaping Filter Parameters

TX Filter: Root Raised Cos

Alpha: 0.50

Filter Length: 6

M-PSK (4): 4

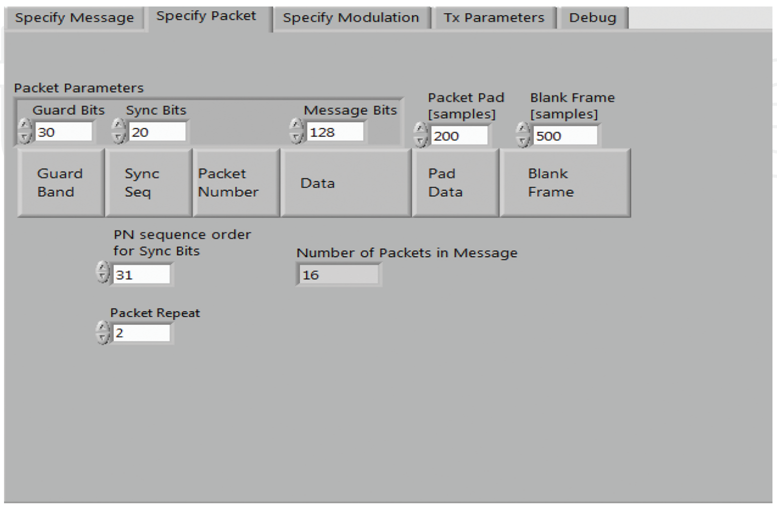
M-QAM (16): 8

Symbol Rate [symbols/sec]: 62.5k

Samples per Symbol: 8

Figure 6. Specify modulation window.

Specify packet window is used to set the packet parameters such as guard bits, sync bits, message bits and packed pad (samples). Message bits need to be set 128 bits for random data, text and 4096 bits for image. The PN sequence order for sync bits is also to be specified in the window as shown in **Figure 7**.

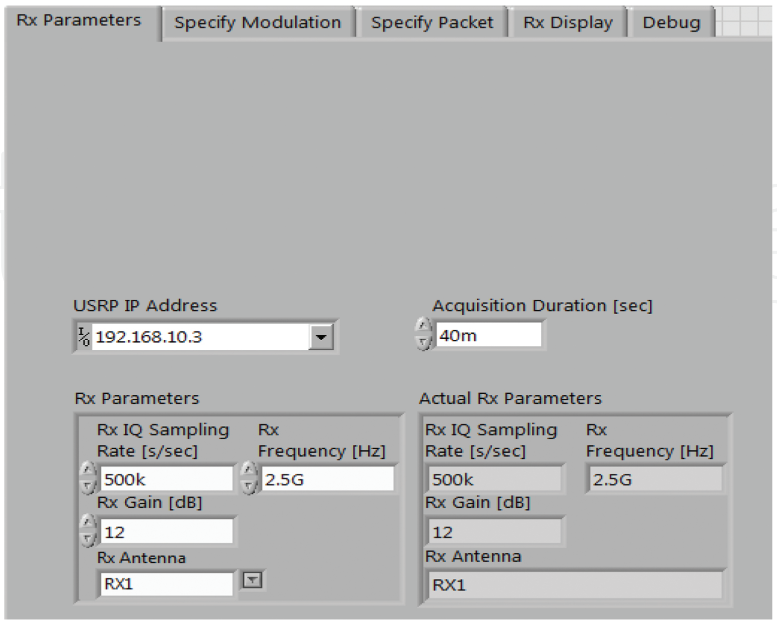


Packet Parameters

Guard Bits 30
Sync Bits 20
Message Bits 128
Packet Pad [samples] 200
Blank Frame [samples] 500
PN sequence order for Sync Bits 31
Number of Packets in Message 16
Packet Repeat 2

Figure 7. Specify packet window.

Now at the receiver end we need to receive the sent message for that the receiver must be set to certain parameters as shown in **Figure 8**. The front panel of the receiver looks as shown in **Figure 8**; select Rx parameter window, specify the IP address, Rx parameter sand acquisition duration [s].



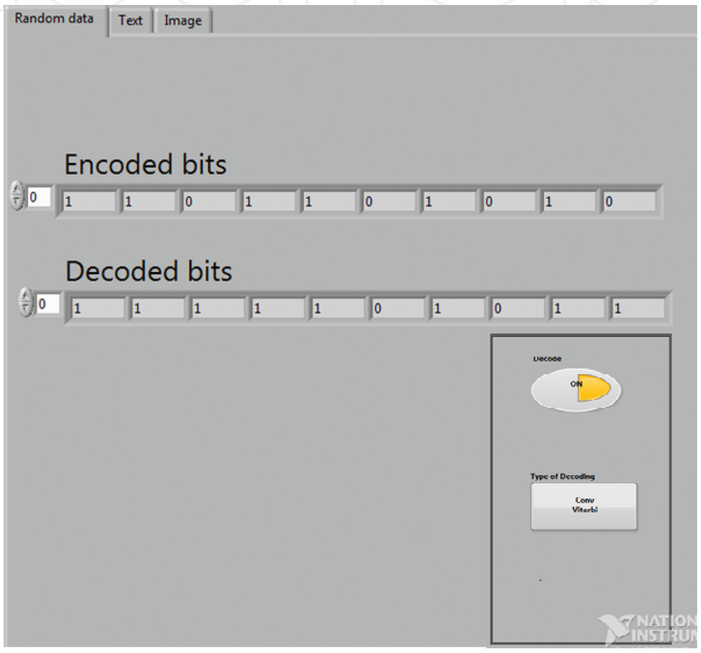
USRP IP Address 192.168.10.3
Acquisition Duration [sec] 40m

Rx Parameters
Rx IQ Sampling Rate [s/sec] 500k
Rx Frequency [Hz] 2.5G
Rx Gain [dB] 12
Rx Antenna RX1

Actual Rx Parameters
Rx IQ Sampling Rate [s/sec] 500k
Rx Frequency [Hz] 2.5G
Rx Gain [dB] 12
Rx Antenna RX1

Figure 8. Front panel of USRP receiver/Rx parameter.

Rx parameter is used to select receiver Rx parameter. USRPIP address for the receiver is specified in the IP address of the receiver device. Rx IQ sampling rate [s/s] is set to 500k to provide the required sampling rate. Rx frequency [Hz] is used to set the frequency to2.5G. Rx gain [dB] is set to12, same as gain for the Tx. Rx antenna field is used to select the Rx antenna as Rx1. In the field acquisition duration [s], set duration 40 m for random data and text, 56 m for image. Now one must select Rx display as per selected specify message in the transmitter. If text is selected at transmitter, then text should be selected at receiver similar to random data and image as shown in **Figures 9–11**.



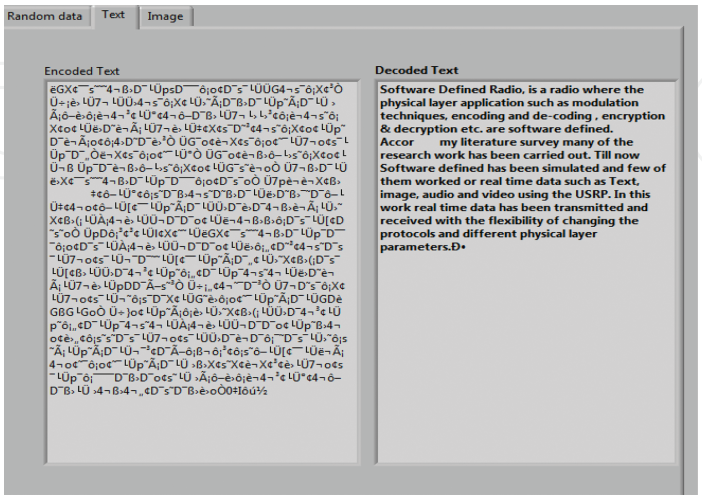
Encoded bits

0	1	1	0	1	1	0	1	0	1	0

Decoded bits

0	1	1	1	1	1	0	1	0	1	1

Figure 9. Rx display/random data Rx.



Encoded Text

```
ëGX¢¯¯s˜˜˜4¬ß›D¯ ˪ÜpsD¯¯¯ô¡o¢D¯s¯ ˪ÜÜG4¬s¯ô¡X¢³Ò
Ü÷¡è› ˪Ü7¬ ˪ÜÜ›4¬s¯ô¡X¢ ˪Ü›˜Ã¡D¯ß›D¯ ˪Üp˜Ã¡D¯ ˪Ü ›
Ã¡ô–è›ô¡è¬4¬³¢ ˪Ü°¢4¬ô–D¯ß› ˪Ü7¬ ˪› ˪›³¢ô¡è¬4¬s˜ô¡
X¢o¢ ˪Üë›D˜è¬Ã¡ ˪Ü7¬è› ˪Ü‡¢X¢s¯D˜³¢4¬s˜ô¡X¢o¢ ˪Üp˜
D¯è¬Ã¡o¢ô¡4›D˜D¯è›³Ò ÜG¯o¢è¬X¢s¯ô¡o¢˜¯ ˪Ü7¬o¢s¯ ˪
Üp¯D¯„Òë¬X¢s¯ô¡o¢˜¯ ˪Ü°Ò ÜG¯o¢è¬ß›ô– ˪›s˜ô¡X¢o¢ ˪
Ü¬ß Üp¯D¯è¬ß›ô– ˪›s˜ô¡X¢o¢ ˪ÜG¯s˜è¬oÒ Ü7¬ß›D¯ ˪Ü
ë›X¢¯¯s˜˜˜4¬ß›D¯ ˪Üp¯D¯¯¯ô¡o¢D¯s¯oÒ Ü7pè¬è¬X¢ß›
            ‡¢ô– ˪Ü°¢ô¡s˜D¯ß›4¬s˜D˜ß›D¯ ˪Üë›D˜ß›¯˜D¯ô– ˪
Ü‡¢4¬o¢ô– ˪Ü[¢¯¯ ˪Üp˜Ã¡D¯ ˪ÜÜ›D¯è›D¯4¬ß›è¬Ã¡ ˪Ü›˜
X¢ß›(¡ ˪ÜÀ¡4¬è› ˪ÜÜ¬D¯D¯o¢ ˪Üë¬4¬ß›ß›ô¡D¯s¯ ˪Ü[¢D
˜s˜oÒ ÜpDô¡³¢³¢ ˪ÜI¢X¢˜˜ ˪ÜëGX¢¯¯s˜˜˜4¬ß›D¯ ˪Üp¯D¯¯
¯ô¡o¢D¯s¯ ˪ÜÀ¡4¬è› ˪ÜÜ¬D¯D¯o¢ ˪Üë›ô¡„¢D˜³¢4¬s˜D¯s
¯ ˪Ü7¬o¢s¯ ˪Ü¬¯D¯˜˜ ˪Ü[¢¯¯ ˪Üp˜Ã¡D¯„¢ ˪Ü›˜X¢ß›(¡D¯s¯
 ˪Ü[¢ß› ˪ÜÜ›D¯4¬³¢ ˪Üp˜ô¡„¢D¯ ˪Üp¯4¬s˜4¬ ˪Üë›D˜è¬
Ã¡ ˪Ü7¬è› ˪ÜpDD¯Ã–s˜³Ò Ü÷¡„¢4¬˜¯D¯³Ò Ü7¬D˜s¯ô¡X¢
 ˪Ü7¬o¢s¯ ˪Ü¬˜ô¡s¯D¯X¢ ˪ÜG˜è›ô¡o¢˜¯ ˪Üp˜Ã¡D¯ ˪ÜGDè
GßG ˪GoÒ Ü÷}o¢ ˪Üp˜Ã¡ô¡è› ˪Ü›˜X¢ß›(¡ ˪ÜÜ›D¯4¬³¢ ˪Ü
p˜ô¡„¢D¯ ˪Üp¯4¬s˜4¬ ˪ÜÀ¡4¬è› ˪ÜÜ¬D¯D¯o¢ ˪Üp˜ß›4¬
o¢è›„¢ô¡s˜s˜D¯s¯ ˪Ü7¬o¢s¯ ˪ÜÜ›D¯è¬D¯ô¡˜¯D¯s¯ ˪Ü›˜ô¡s
˜Ã¡ ˪Üp˜Ã¡D¯ ˪Ü¬¯³¢D¯Ã–ô¡ß¬ô¡³¢ô¡s˜ô– ˪Ü[¢¯¯ ˪Üë¬Ã¡
4¬o¢˜¯ô¡o¢˜¯ ˪Üp˜Ã¡D¯ ˪Ü ›ß›X¢s˜X¢è¬X¢³¢è› ˪Ü7¬o¢s
¯ ˪Üp¯ô¡¯¯¯¯D¯ß›D¯o¢s˜ ˪Ü ›Ã¡ô–è›ô¡è¬4¬³¢ ˪Ü°¢4¬ô–
D¯ß› ˪Ü ›4¬ß›4¬„¢D¯s˜D¯ß›è›oÒ0‡Iôú½
```

Decoded Text

```
Software Defined Radio, is a radio where the
physical layer application such as modulation
techniques, encoding and de-coding , encryption
& decryption etc. are software defined.
Accor       my literature survey many of the
research work has been carried out. Till now
Software defined has been simulated and few of
them worked or real time data such as Text,
image, audio and video using the USRP. In this
work real time data has been transmitted and
received with the flexibility of changing the
protocols and different physical layer
parameters.Ð•
```

Figure 10. Rx display/text Rx.

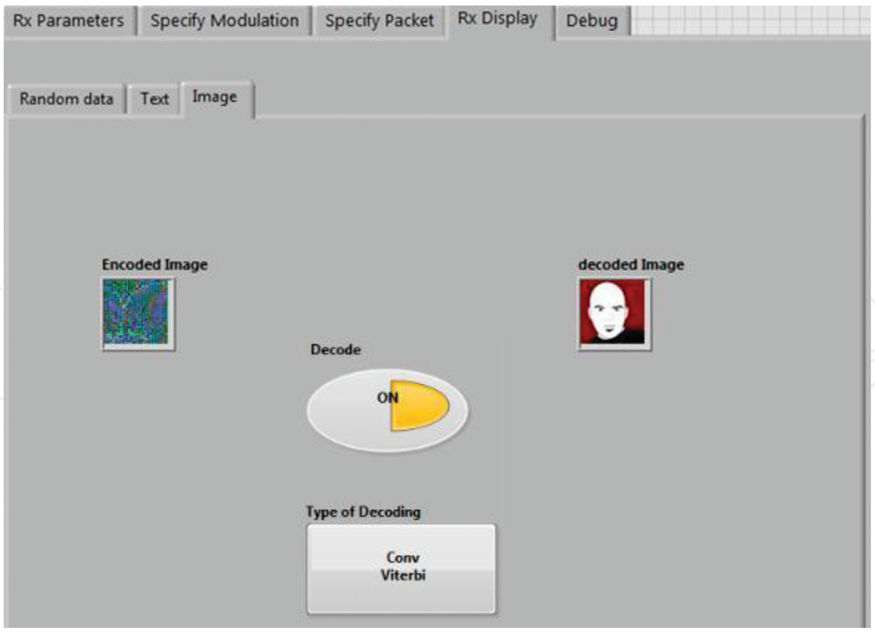
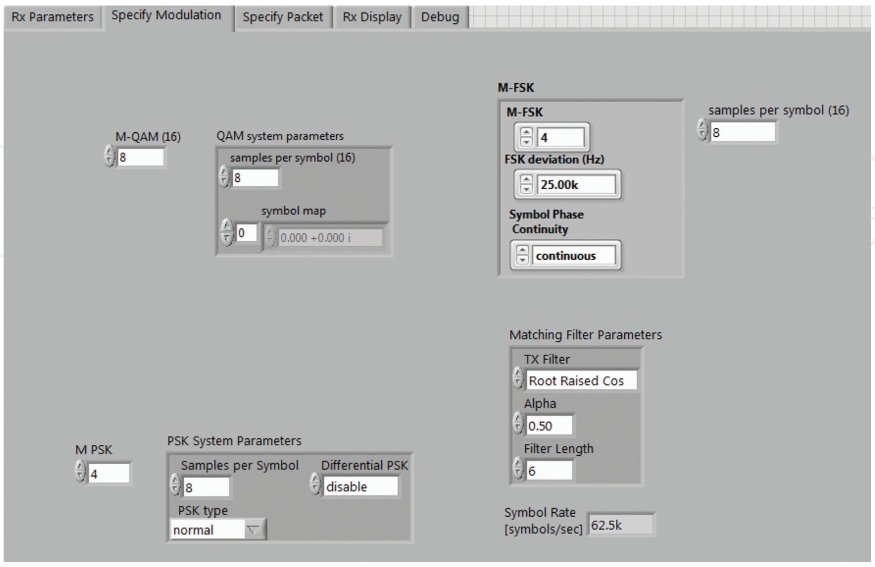


Figure 11. Rx display/image.

Random data: Click on random data in order to receive random data. Encoded bits in this window indicate the bits which are encoded. Decoded bits indicate the bits that are decoded from the transmitted data. Decode button is used to decode the received data. Type of decoding field is used to select the type of decoding required either convolution viterbi or turbo coding which is selected at the transmitter.

For receiving the text, we need to select the text field. Encoded text field indicates the encoded text. Decoded text field indicates the decoded text.



M-QAM (16): 8

QAM system parameters
samples per symbol (16): 8
symbol map: 0, 0.000 +0.000 i

M-FSK
M-FSK: 4
FSK deviation (Hz): 25.00k
Symbol Phase Continuity: continuous

samples per symbol (16): 8

Matching Filter Parameters
TX Filter: Root Raised Cos
Alpha: 0.50
Filter Length: 6

M PSK: 4

PSK System Parameters
Samples per Symbol: 8
Differential PSK: disable
PSK type: normal

Symbol Rate [symbols/sec]: 62.5k

Figure 12. Specify demodulation window.

For receiving the image, we need to select the image field. Encoded image field indicates the encoded image. Decoded image field indicates the decoded image. Decode button is used to decode the received data. Type of decoding needs to be set to either convolution viterbi or turbo coding which is selected at the transmitter. Now, select the specify demodulation window, as shown in **Figure 12**, all the parameters to be set exactly the same as transmitter modulation.

Matching filter parameters: Set the matching filter parameters as shown in **Figure 12**:

a. Tx filter: Select the type of filter required raised cosine filter or root raised cosine or none.

b. Alpha: Set the alpha to 0.50.

c. Filter length: Set filter length to 6.

Samples per symbol: Set samples per symbol to 8. **Symbol rate [symbols/s]. M-FSK:** Set the FSK parameters as shown in **Figure 13**:

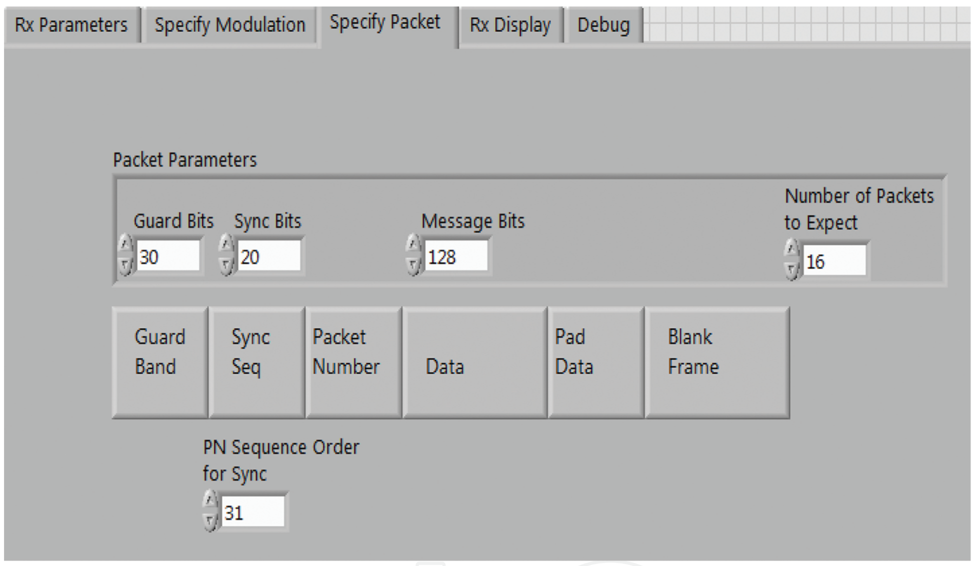

Packet Parameters
Guard Bits 30
Sync Bits 20
Message Bits 128
Number of Packets to Expect 16
PN Sequence Order for Sync 31

Figure 13. Receiver/specify packet window.

a. M-FSK: Set M-FSK to (any M-ary value)

b. M-PSK: Set M-PSK to (any M-ary value)

c. M-QAM: Set M-QAM to (any M-ary value)

d. FSK deviation [Hz]: Set FSK deviation to 25.00 k

e. Symbol phase continuity: Select symbol phase continuity to continuous.

Now, select the specify packet window from the receiver front panel, as shown in **Figure 13**.

Specify packet specifies the packet same as transmitter, that is, 128 for random data and text, for image 4096. Message bits field holds the number of bits, same as in the transmitter. Number of packets to be expected must be entered in this window which is the same as in the transmitter. Now, select output parameters window, which shows the constellation diagram, eye diagram and BER Vs E_b/N_o with respective received data as shown in **Figure 14**.

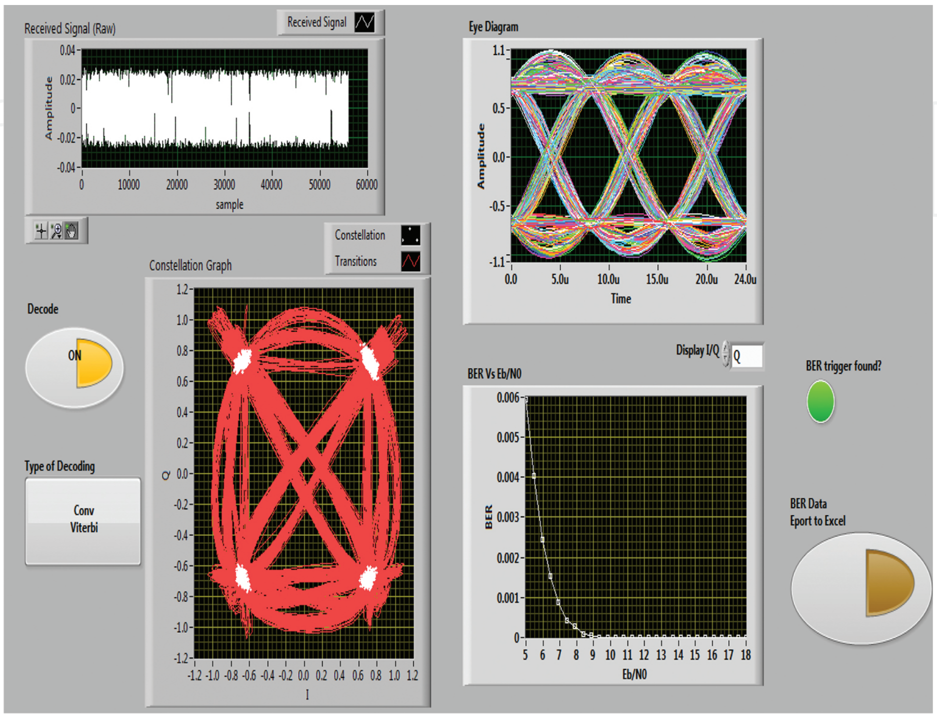


Figure 14. Output parameters.

Constellation graph is used to indicate the constellation of the received bits with respect to data. Eye diagram indicates the eye diagram of the received bits with respect to data. BER Vs E_b/N_o indicates the BER Vs E_b/N_o of the received bits with respect to data. BER Data Export to Excel: Click on BER Data Export to Excel Button it will export BER Vs E_b/N_o data to Excel sheet. Decode: Press the decode button to decode the received data. Type of decoding: Select the type of decoding required either convolution viterbi or turbo coding which is selected at the transmitter.

2. Conclusion

USRP can be effectively combined to create an SDR transceiver. The LabVIEW/USRP combination presents an opportunity to enhance communications education by enabling a low-cost, hands-on approach with live signals for realistic, real-world demonstrations, laboratory exercises, capstone design projects and cutting-edge research.

The universal software radio peripheral family of products is a popular platform for hardware-based research and test bed validations conducted by universities in the software-defined radio and cognitive radio (CR) fields. The USRP offers a simpler, scalable and easier to use combined platform that will both broaden the accessibility of the technology and platform for hands-on applications and spur further adoption and use within university communication systems classrooms, teaching laboratories and their natural follow-on coursework (e.g., SDR, CR, digital communications, wireless communications and satellite communications).This chapter describes SDR system built using LabVIEW and testing the output using real-time data. Through this chapter, we have tried to cover emerging SDR standards and the technologies being used to specify and support them. We have proposed expanding the SDR definition and discussed the issues pertaining to the design of a multi-band flexible receiver and a linearized transmitter using broadband quadrature techniques. Here, we have described the case study of the SDR by designing an SDR generic transceiver in LabVIEW highlighting the multi-modulation approach for the SDR. The design uses forward error correction codes namely convolution codes and turbo codes for enhanced security for the data being transmitted. The proposed design is entirely reconfigurable in nature and supports multiple M-ary modulation schemes which can be changed accordingly by the user any time during the runtime. The biggest advantage of this design is that we have used phase tracking for identification of the constellation points. The analysis of the case study proves that turbo coding gives a much improved and better minimization of the data errors than the convolution coding.

Author details

Nikhil Marriwala[1*], Om. Prakash. Sahu[2] and Anil Vohra[3]

*Address all correspondence to: nikhilmarriwala@gmail.com

1 Electronics and Communication Engineering Department, University Institute of Engineering and Technology, Kurukshetra University, Kurukshetra, Haryana, India

2 Electronics and Communication Engineering Department, NIT, Kurukshetra, Haryana, India

3 Electronics Science Department, Kurukshetra University, Kurukshetra, Haryana, India

References

[1] J. Mitola, "Software and DSP in Radio," IEEE Communications Magazine, Volume: 38, Issue: 2, Feb 2000, ISSN 0163-6804.

[2] M. N. O. Sadiku and C. M. Akujuobi, "Software-defined radio: a brief overview," *IEEE Potentials*, vol. 23 no. 4, pp. 14–15, 2004.

[3] J. C. Lyke, C. G. Christodoulou, G. A. Vera and A. H. Edwards, "An introduction to reconfigurable systems," *Proc. IEEE*, vol. 103, no. 3, pp. 291–317, 2015.

[4] N. Marriwala, O. P. Sahu and A. Vohra, "Novel design of a low cost flexible transceiver based on multistate digitally modulated signals using Wi-Fi protocol for software defined radio," *Wirel. Pers. Commun.*, 2016, 87: 1265. doi:10.1007/s11277-015-3052-4

[5] N. Marriwala, O. P. Sahu and A. Vohra, "Design of a hybrid reconfigurable Software Defined Radio transceiver based on frequency shift keying using multiple encoding schemes," *Egypt. Informatics J.*, vol. 17, no. 1, pp. 89–98, 2015.

[6] N. Shahin, N. J. Lasorte, S. A. Rajab and H. H. Refai, "802.11g channel characterization utilizing labview and NI-USRP," *Conf. Rec. – IEEE Instrum. Meas. Technol. Conf., The Depot Minneapolis*, MN, USA, pp. 753–756, 2013.

[7] "A mathematical theory of communication," C. E. Shannon, System, vol. 27, no. July, 1928, pp. 379–423, 1948.

[8] N. Marriwala, "LabVIEW based design implementation of M-PSK transceiver using multiple forward error correction coding technique for software defined radio applications," *J. Electr. Electron. Eng.*, vol. 2, no. 4, p. 55, 2014.

[9] Y. Z. Y. Zhu and C. Chakrabarti, "Architecture-Aware LDPC code design for multiprocessor software defined radio systems," *IEEE Trans. Signal Process.*, vol. 57, no. 9, pp. 3679–3692, 2009.

[10] H. Shehab and W. Ismail, "The development & implementation of Reed Solomon codes for OFDM using software-defined radio platform," *Int. J. Comput. Sci. Commun.*, vol. 1, no. 1, pp. 129–136, 2010.

[11] P. F. Morlat, A. Luna, X. Gallon, G. Villemaud, J. M. Gorce and C. Insa-lyon, "Structure and implementation of a SIMO multi-standard multi-channel SDR receiver," HAL archives open, inria-00412054, version 1, pp. 283–286, 2008.

[12] P. B. Kenington, *TEAM LinG*. Boston, London: Artech House, 2005.

[13] M. Imran Anwar, S. Virtanen and J. Isoaho, "A software defined approach for common baseband processing," *J. Syst. Archit.*, vol. 54, no. 8, pp. 769–786, 2008.

[14] T. Javornik, M. Mohorčič, A. Švigelj, I. Ozimek and G. Kandus, "Adaptive coding and modulation for mobile wireless access via high altitude platforms," *Wirel. Pers. Commun.*, vol. 32, no. 3–4, pp. 301–317, 2005.

[15] C. B. Haskins and W. P. Millard, "Multi-band software defined radio for spaceborne communications, navigation, radio science and sensors," *IEEEAC Conf. IEEE*, vol. 5, pp. 1–9, 2010.

[16] D. Wu, J. Eilert and D. Liu, "Implementation of a high-speed MIMO soft-output symbol detector for software defined radio," *J. Signal Process. Syst.*, vol. 63, no. 1, pp. 27–37, 2009.

[17] I. Hatai and I. Chakrabarti, "FPGA implementation of a digital FM modem for SDR architecture," *Comput. Devices Commun. 2009. CODEC 2009. 4th Int. Conf.*, Kolkata, India, no. August, 2009.

[18] G. Baldini, T. Sturman, A. R. Biswas and M. Street, "Security aspects in software defined radio and cognitive radio networks: a survey and a way ahead," *IEEE Commun. Surv. TUTORIALS*, vol. 14, no. 2, pp. 355–379, 2012.