HARDWARE IMPLEMENTATION OF NARROWBAND SDR (SOFTWARE DEFINED RADIO) WAVEFORM

By

Ahmed Ayub Butt

Ehtasham Naseer

Faheem Anjum

Submitted to the Faculty of Department of Electrical Engineering, Military College of Signals, National University of Sciences and Technology, Islamabad in partial fulfillment for the requirements of B.E Degree in Electrical Engineering JUNE 2016

Abstract

HARDWARE IMPLEMENTATION OF NARROWBAND SDR (SOFTWARE DEFINED RADIO) WAVEFORM

Our project is based on Software Defined Radio technology which aims to take advantage of programmable hardware modules to build Open architecture based radio system software. This leads to device flexibility, software portability and system upgradability.

This project focuses to develop VHF/UHF Narrowband and Wideband waveforms for existing SDR platforms. The narrowband waveform based on frequency modulation (FM) designed for audio transmission and OFDM Wideband waveform designed for data transmission coded in C++ and Python are implemented on the platform using Labview software.

The project envisions creating a standalone Software Defined Radio device dedicated to the transmission and reception of the indigenously developed VHF/UHF Narrowband and Wideband waveforms. The project will help to intrinsically maximize the benefits of SDR by providing waveforms to manufacturers building commercial base stations, mobiles and small form-factor handheld radios to allow them to reap the practical benefits of reusable, maintainable code. It will also pave the path for the further development of the telecommunication and information technology industry in Pakistan, making them capable to offer system based on latest wireless reconfigurable technologies.

CERTIFICATE FOR CORRECTNESS AND APPROVAL

It is certified that the work contained in the thesis for Hardware implementation of narrowband SDR (Software Defined Radio) waveform Ahmed Butt, Ehtashaam Naseer & Faheem Anjum under the supervision of Lec Lt-Col Adnan Rashdi for partial fulfillment of Degree of Bachelor of Electrical Engineering is correct and approved.

Approved by

Lec Lt-Col Adnan Rashdi

Electrical Engineering Dept.
MCS

DECLARATION

No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent.
To our parents, without whose unflinching support and unstinting cooperation, a work of this magnitude would not have been possible.

ACKNOWLEDGEMENT

We bow in gratitude to Allah Almighty for giving us strength and knowledge to accomplish this task as nothing happens without his will. The group is indebted by the immense help and moral support given to us by our parents as it would not have been possible without their prayers. The team also likes to thank our project supervisor, Lec Lt Col Adnan Rashdi, without his support and encouragement, it would not have been impossible to complete this project.

Contents

INTRODUCTION

USRP (Universal Software Defined Radio) Software Defined Radio Fundamentals

What is Software Defined Radio?

The Wireless Innovation Forum defines Software Defined Radio (SDR) as: “Radio in which some or all of the physical layer functions are software defined.” 1 SDR refers to the technology wherein software modules running on a generic hardware platform are used to implement radio functions. Combine the NI USRP hardware with LabVIEW software for the flexibility and functionality to deliver a platform for rapid prototyping involving physical layer design, wireless signal record & playback, signal intelligence, algorithm validation, and more.

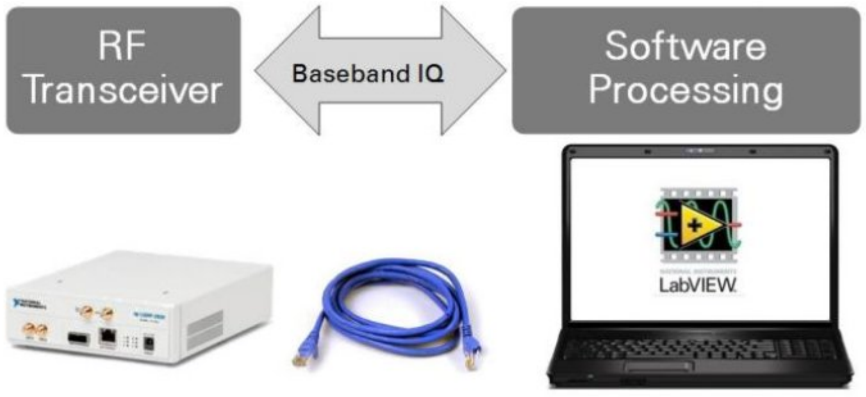


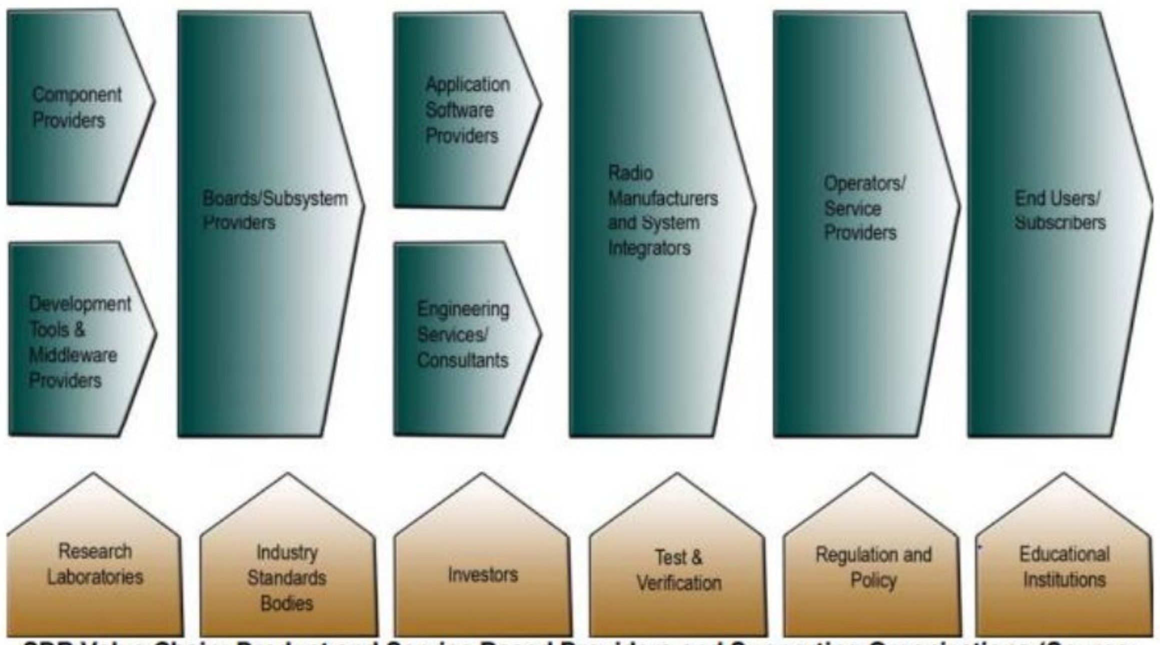
Figure 2. Simplified Overview of a SDR Setup Built Around an NI USRP

Software Defined Radio – Value Chain:

The time is now to engage SDR at all levels of the chain

The benefits and anticipated opportunities for SDR technology are having a significant impact on the wireless industry’s value chain. This chain consists of product-based and service-based providers, with value added at each stage, ultimately resulting in SDR end products and services that meet the needs of the end users and subscribers.

Throughout the chain, the providers may be supported by external organizations such as educational institutions, research laboratories, industry standards bodies, investors, tests & verification and government. These supporting organizations provide critical input as development progresses through the chain, ultimately reaching the end user. The detail of the chain and the relationship within the context of the SDR Forum membership is outlined below


Component Providers
Development Tools & Middleware Providers
Boards/Subsystem Providers
Application Software Providers
Engineering Services/ Consultants
Radio Manufacturers and System Integrators
Operators/ Service Providers
End Users/ Subscribers
Research Laboratories
Industry Standards Bodies
Investors
Test & Verification
Regulation and Policy
Educational Institutions

SDR Value Chain: Product and Service Based Providers and Supporting Organizations (Source: SDR Forum 2005 Year Book)

Digital Communication System Fundamentals

A typical digital communication system includes a transmitter, a receiver, and a communication channel. Figure 3 illustrates the general components of a digital communication system. The transmitter, shown as blocks on the top row, contains blocks for source and channel coding, modulation, simulating real-world signal impairments, and up conversion. The receiver, which includes the blocks in the bottom row, has blocks for down conversion, matched filtering, equalization, demodulation, channel decoding and source decoding. Refer to the NI LabVIEW Modulation Toolkit online Help for more information about measurement and visualization tools for digital communication systems.

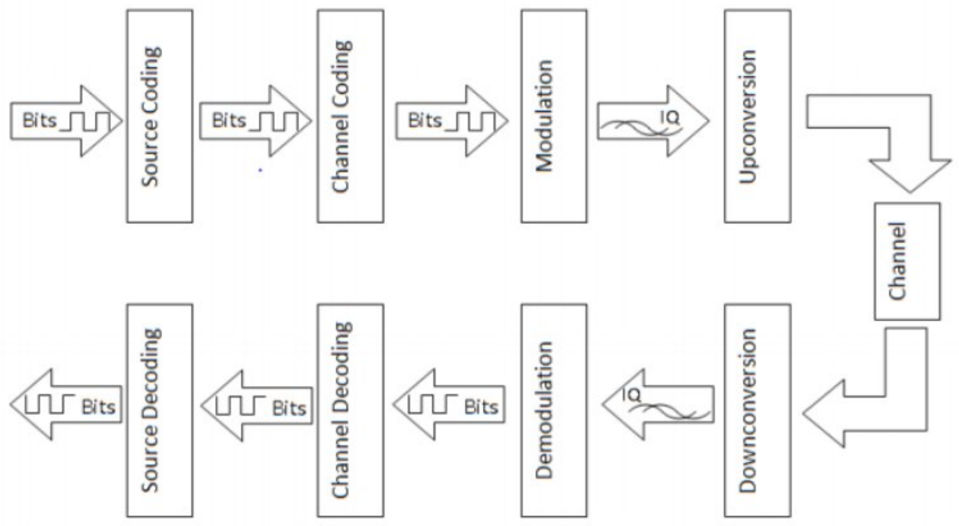


Figure 3. Digital Communication System Block Diagram

What is USRP

Universal Software Radio Peripheral (**USRP**) is a range of software-defined radios designed and sold by Ettus Research and its parent company, National Instruments. Developed by a team led by Matt Ettus, the USRP product family is intended to be a comparatively inexpensive hardware platform for software radio, and is commonly used by research labs, universities, and hobbyists.

Most USRPs connect to a host computer through a high-speed link, which the host-based software uses to control the USRP hardware and transmit/receive data. Some USRP models also integrate the general functionality of a host computer with an embedded processor that allows the USRP device to operate in a stand-alone fashion.

- Tunable RF transceiver with frequency options from 50 MHz to 6 GHz
- Up to 20 MHz of real-time bandwidth with plug-and-play MIMO support
- Option for an integrated GPS receiver for better frequency accuracy and synchronization
- Ready-to-use hands-on laboratory manuals for RF and communications courses
- 1 Gbit Ethernet connection to host

The NI USRP™ (Universal Software Radio Peripheral) is an affordable, flexible radio that turns a standard PC into a wireless prototyping platform. Paired with NI LabVIEW software, NI USRP transceivers offer a powerful system to get you up and running quickly.

USRP Hardware

The NI USRP connects to a host PC to act as a software-defined radio. Incoming signals attached to the standard SMA connector are mixed down using a direct-conversion receiver (DCR) to baseband I/Q components, which are sampled by a 2-channel, 100 MS/s, 14-bit analog-to-digital converter (ADC). The digitized I/Q data follows parallel paths through a digital downconversion (DDC) process that mixes, filters, and decimates the input 100 MS/s signal to a user-specified rate. The downconverted samples, when represented as 32-bit numbers (16 bits each for I and Q), are passed to the host computer at up to 20 MS/s over a standard Gigabit Ethernet connection.

For transmission, baseband I/Q signal samples are synthesized by the host computer and fed to the USRP-2920 at up to 20 MS/s over Gigabit Ethernet when represented with 32-bits (16-bits each for the I and Q components). The USRP hardware interpolates the incoming signal to 400 MS/s using a digital upconversion (DUC) process and then converts the signal to analog with a dual-channel, 16-bit digitalto-analog converter (DAC). The resulting analog signal is then mixed up to the specified carrier frequency.

An available 8-bit mode, in which 16-bits total are used to represent the I and Q values of a downconverted sample or sample to be upconverted, can enable a transfer rate of up to 40 MS/s over the Gigabit Ethernet connection between the host PC and the USRP.

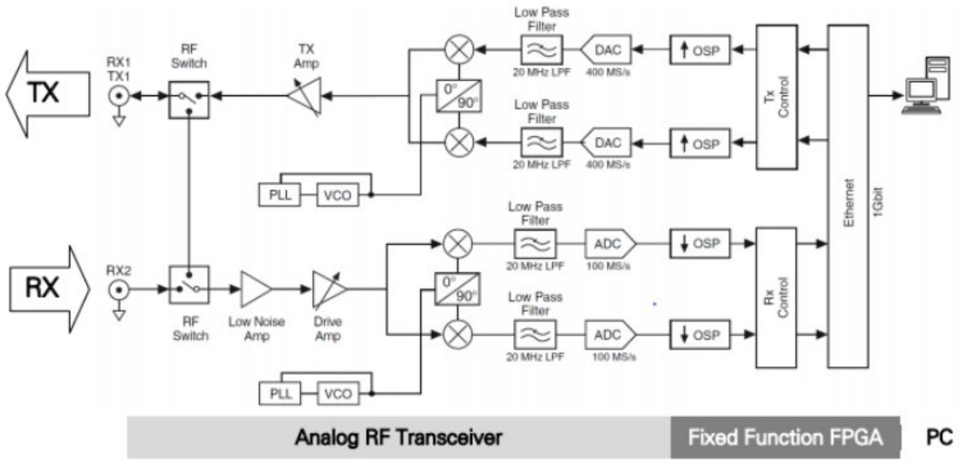


Figure 4. USRP Block Diagram

Benefits for Educators

The NI USRP platform offers a new approach to RF and communications education, which has traditionally been limited to a focus on mathematical theory. With NI USRP and LabVIEW, students gain hands-on experience exploring a working communications system with live signals to gain a better understanding of the link between theory and practical implementation.

Integrated GPS Receiver With NI USRP-1

The NI USRP1 devices are equipped with a GPS-disciplined 10 MHz oven-controlled crystal oscillator (OCXO) reference clock. The OCXO reference clock is 100 times more accurate than a standard temperature compensated crystal oscillator. The GPS disciplining delivers improved frequency accuracy and synchronization capabilities.

Introduction to the USRP board

These days, when we talk about the GNU Radio, the Universal Software Radio Peripheral (USRP) board has become an indispensable hardware component. It is developed by Matt wholly for the GNU Radio users. Basically, the USRP is an integrated board which incorporates AD/DA converters, some forms of RF front end, and an FPGA which does some important but computationally expensive pre-processing of the input signal. The USRP is low-cost and high speed, which is the best choice for a GNU Radio user to implement some real time applications. We could purchase the USRP boards from Ettus. A USRP board consists of one mother board and up to four daughter boards. The price for the mother board is $450 and basic daughterboards cost $50 each

A 'data sheet' of the USRP board

This section introduces the hardware parts on the USRP board. It should be emphasized that the characteristics of those hardware parts are very important. They will influence your radio design and software programming extensively. You have to follow the constraints imposed by the hardware and memorizing some of them would be useful. So please read this section carefully

AD / DA Converters

There are 4 high-speed 12-bit AD converters. The sampling rate is 64M samples per second. In principle, it could digitize a band as wide as 32MHz. The AD converters can bandpass-sample signals of up to about 150MHz, though. If we sample a signal with the IF larger than 32MHz, we introduce aliasing and actually the band of the signal of interest is mapped to some places between -32MHz and 32MHz. Sometimes this can be useful, for example, we could listen to the FM stations without any RF front end. The higher the frequency of the sampled signal, the more the SNR will be degraded by jitter. 100MHz is the recommended upper limit. The full range on the ADCs is 2V peak to peak, and the input is 50 ohms differential. This is 40mW, or 16dBm. The DAC clock frequency is 128 MS/s, so Nyquist frequency is 64MHz. However, we will probably want to stay below about 50MHz or so to make filtering easier. So a useful output frequency range is DC to about 50MHz. The DACs can supply 1V peak to a 50 ohm differential load, or 10mW (10dBm). There is also PGA used after the DAC, providing up to 20dB gain. Note that the PGAs on both RX and TX paths are programmable

The daughter boards

On the mother board there are four slots, where you can plug in up to 2 RX daughter boards and 2 TX daughter boards. The daughter boards are used to hold the the RF receiver interface or tuner and the RF transmitter. There are slots for 2 TX daughter boards, labeled TXA and TXB, and 2 corresponding RX daughter boards, RXA and RXB. Each daughter board slot has access to 2 of the 4 high-speed AD / DA converters (DAC outputs for TX, ADC inputs for RX). This allows each daughter board which uses real (not IQ) sampling to have 2 independent RF sections, and 2 antennas (4 total for the system). If complex IQ sampling is used, each board can support a single RF section, for a total of 2 for the whole system. 4 Tutorial 4 - The USRP Board We can see there are two SMA connectors on each daughter board. We normally use them to connect the input or output signals. There several kinds of daughter boards available now: Basic daughter boards. Nothing fancy on it.

Two SMA connectors are used to connect external tuners or signal generators. We can treat it as an entrance or an exit for the signal without affecting it. Some form of external RF front end is required. TVRX daughter boards. With Microtune 4937 Cable Modem tuner equipped. This is a receiveonly daughter board. The RF frequency ranges from 50MHz to 800MHz, with an IF bandwidth of 6MHz. What you need is just an antenna if your radio application is within this range, such as FM or TV detection. DBSRX daughter boards. Similar with the TVRX boards. This is also receive-only. The RF frequency ranges from 800MHz to 2.4GHz. Some new daughter boards are being developed, especially including the tranceiver daughter boards. They will be on sale soon and will bring us great convenience.

Products

Networked series

The USRP N200 and USRP N210 are high-performance USRP devices that provide higher dynamic range and higher bandwidth than the bus series. Using a Gigabit Ethernetinterface, the devices in the Networked Series can transfer up to 50 MS/s of complex, baseband samples to/from the host. This series uses a dual, 14-bit, 100 MS/S ADC and dual 16-bit, 400 MS/s DAC. This series also provides a MIMO expansion port which can be used to synchronize two devices from this series. This is the recommended solution for MIMO systems.

USRP N200

- A Xilinx Spartan-3A DSP 1800 FPGA
- Gigabit Ethernet interface
- Dual 100 MS/s, 14-bit, analog-to-digital converter
- Dual 400 MS/s, 16-bit, digital-to-analog converter
- Flexible Clocking and Synchronization
  - External Inputs for 10 MHz and 1 PPS signals (SMA)
  - Optional GPS Disciplined Oscillator
  - Ettus Research MIMO Cable that can be used to synchronize two USRP devices (sold separately)

USRP N210

- A Xilinx Spartan-3A DSP 3400 FPGA
- Gigabit Ethernet interface
- Dual 100 MS/s, 14-bit, analog-to-digital converter
- Dual 400 MS/s, 16-bit, digital-to-analog converter
- Flexible Clocking and Synchronization
  - External Inputs for 10 MHz and 1 PPS signals (SMA)
  - Optional GPS Disciplined Oscillator
  - Ettus Research MIMO Cable that can be used to synchronize two USRP devices (sold separately)
  - Support for timed commands and LO alignment with the SBX daughterboard

Bus series

All products in Ettus Research Bus Series use a USB 2.0 or USB 3.0 interface to transfer samples to and from the host computer. These are designed for applications that do not require the higher bandwidth and dynamic range provided by the Network Series(USRP N200 and USRP N210).

USRP1

The USRP1 is the original USRP product and consists of:

- Four high-speed analog-to-digital converters, each capable of 64 MS/s at a resolution of 12-bit, 85 dB SFDR (AD9862).
- Four high-speed digital-to-analog converters, each capable of 128 MS/s at a resolution of 14-bit, 83 dB SFDR (AD9862).
- An Altera Cyclone EP1C12Q240C8 FPGA.

- A Cypress EZ-USB FX2 High-speed USB 2.0 controller.
- Four extension sockets (2 TX, 2 RX) in order to connect 2–4 daughterboards.
- 64 GPIO pins available through four BasicTX/BasicRX daughterboard modules (16 pins each).
- Glue logic.

USRP B100

The B100, introduced in October 2011, replaces the USRP as the basic Software Defined Radio offering from Ettus Research. The features of the B100 are:

- USB 2.0 interface
- Xilinx Spartan 3A-1400 FPGA
- Compatibility with the entire daughterboard family
- Fully supported by UHD drivers
- Dual 64 MS/s 12-bit ADCs
- Dual 128 MS/s 14-bit DACs
- Onboard TCXO for precise frequency control
- 10 MHz and 1 PPS inputs for external references
- Flexible clocking from 10 MHz to 64 MHz
- 8 MHz of RF bandwidth with 16 bit samples
- 16 MHz of RF bandwidth with 8 bit samples

Software LAB-VIEW

From the inception of an idea to the commercialization of a widget, NI’s unique platform-based approach to engineering and science applications has driven progress across a wide variety of industries. Central to this approach is LabVIEW, a development environment designed specifically to accelerate the productivity of engineers and scientists. With a graphical programming syntax that makes it simple to visualize, create, and code engineering systems.

LabVIEW is unmatched in helping engineers translate their ideas into reality, reduce test times, and deliver business insights based on collected data. From building smart machines to ensuring the quality of connected devices, LabVIEW has been the preferred solution to create, deploy, and test the Internet of Things for decades.

Why LABVIEW

LabVIEW increases productivity by abstracting low-level complexity and integrating all of the technology you need into a single, unified development environment, unlike any other text-based alternative. Programming in a unified environment means you don’t have to invest time in building expertise in a variety of tools to accomplish your goal. Instead, you can be confident that elements of your system will fit together seamlessly.

LabVIEW is the cornerstone of NI’s system design platform. Investing in a platform approach gives you the ability to effectively scale your application to meet changing requirements. The integrated software and hardware nature of this platform makes it easier to visualize and implement all of the technology, I/O, math, and models needed for a system. Within LabVIEW, you can use different programming approaches with the latest technologies to describe the way your system works, and then compile to the best target to run your system. This way you can iterate quickly on not just the design, but the implementation, of the systems you need.

Graphical Programming

LabVIEW ties the creation of user interfaces (called front panels) into the development cycle. LabVIEW programs/subroutines are called virtual instruments (VIs). Each VI has three components: a block diagram, a front panel and a connector panel. The last is used to represent the VI in the block diagrams of other, calling VIs. The front panel is built using controls and indicators. Controls are inputs – they allow a user to supply information to the VI. Indicators are outputs – they indicate, or display, the results based on the inputs given to the VI. The back panel, which is a block diagram, contains the graphical source code. All of the objects placed on the front panel will appear on the back panel as terminals. The back panel also contains structures and functions which perform operations on controls and supply data to indicators. The structures and functions are found on the Functions palette and can be placed on the back panel. Collectively controls, indicators, structures and functions will be referred to as nodes. Nodes are connected to one another using wires – e.g. two controls and an indicator can be wired to the addition function so that the indicator displays the sum of the two controls. Thus a virtual instrument can either be run as a program, with the front panel serving as a user interface, or, when dropped as a node onto the block diagram, the front panel defines the inputs and outputs for the node through the connector pane. This implies each VI can be easily tested before being embedded as a subroutine into a larger program.

The graphical approach also allows non-programmers to build programs by dragging and dropping virtual representations of lab equipment with which they are already familiar. The LabVIEW programming environment, with the included examples and documentation, makes it simple to create small applications. This is a benefit on one side, but there is also a certain danger of underestimating the expertise needed for high-quality G programming. For complex algorithms or large-scale code, it is important that the programmer possess an extensive knowledge of the special LabVIEW syntax and the topology of its memory management. The most advanced LabVIEW development systems offer the possibility of building stand-alone applications. Furthermore, it is possible to create distributed applications, which communicate by a client/server scheme, and are therefore easier to implement due to the inherently parallel nature of G.

Benefits

Interfacing to Devices

LabVIEW includes extensive support for interfacing to devices, instruments, cameras, and other devices. Users interface to hardware by either writing direct bus commands (USB, GPIB, Serial) or using high-level, device-specific, drivers that provide native LabVIEW function nodes for controlling the device.

LabVIEW includes built-in support for NI hardware platforms such as Compact DAQ and Compact RIO, with a large number of device-specific blocks for such hardware, the *Measurement and Automation eXplorer* (MAX) and *Virtual Instrument Software Architecture* (VISA) toolsets.

National Instruments makes thousands of device drivers available for download on the NI Instrument Driver Network (IDNet).

Code compilation

In terms of performance, LabVIEW includes a compiler that produces native code for the CPU platform. The graphical code is translated into executable machine code by interpreting the syntax and by compilation. The LabVIEW syntax is strictly enforced during the editing process and compiled into the executable machine code when requested to run or upon saving. In the latter case, the executable and the source code are merged into a single file. The executable runs with the help of the LabVIEW run-time engine, which contains some precompiled code to perform common tasks that are defined by the G language. The run-time engine reduces compilation time and also provides a consistent interface to various operating systems, graphic systems, hardware components, etc. The run-time environment makes the code portable across platforms. Generally, LabVIEW code can be slower than equivalent compiled C code, although the differences often lie more with program optimization than inherent execution speed.

Large libraries

Many libraries with a large number of functions for data acquisition, signal generation, mathematics, statistics, signal conditioning, analysis, etc., along with numerous graphical interface elements are provided in several LabVIEW package options. The number of advanced mathematic blocks for functions such as integration, filters, and other specialized capabilities usually associated with data capture from hardware sensors is enormous. In addition, LabVIEW includes a text-based programming component called MathScript with additional functionality for signal processing, analysis and mathematics. MathScript can be integrated with graphical programming using "script nodes" and uses a syntax that is generally compatible with MATLAB.

Parallel programming

LabVIEW is an inherently concurrent language, so it is very easy to program multiple tasks that are performed in parallel by means of multithreading. This is, for instance, easily done by drawing two or more parallel while loops. This is a great benefit for test system automation, where it is common practice to run processes like test sequencing, data recording, and hardware interfacing in parallel.

Ecosystem

Due to the longevity and popularity of the LabVIEW language, and the ability for users to extend the functionality, a large ecosystem of third party add-ons has developed through contributions from the community. This ecosystem is available on the LabVIEW Tools Network, which is a marketplace for both free and paid LabVIEW add-ons.

User community

There is a low-cost LabVIEW Student Edition aimed at educational institutions for learning purposes. There is also an active community of LabVIEW users who communicate through several e-mail groups and Internet forums.

Home Bundle Edition

National Instrument provides a low cost LabVIEW Home Bundle Edition.

Repositories and libraries

OpenG, as well as LAVA Code Repository (LAVAcr), serve as repositories for a wide range of Open Source LabVIEW applications and libraries. SourceForge has LabVIEW listed as one of the possible languages in which code can be written.

VI Package Manager has become the standard package manager for LabVIEW libraries. It is very similar in purpose to Ruby's RubyGems and Perl's CPAN, although it provides a graphical user interface similar to the Synaptic Package Manager. VI Package Manager provides access to a repository of the OpenG (and other) libraries for LabVIEW.

Related Software

National Instruments also offers a product called Measurement Studio, which offers many of the test, measurement and control capabilities of LabVIEW, as a set of classes for use with Microsoft Visual Studio. This allows developers to harness some of LabVIEW's strengths within the text-based .NET framework. National Instruments also offers LabWindows/CVI as an alternative for ANSI C programmers.

When applications require sequencing, users often use LabVIEW with TestStand test management software, also from National Instruments.

The Ch interpreter is a C/C++ interpreter that can be embedded into LabVIEW for scripting.

The TRIL Centre Ireland BioMobius platform and DSP Robotics' FlowStone DSP also use a form of graphical programming similar to LabVIEW, but are limited to the biomedical and robotics industries respectively.

LabVIEW has a direct node with modeFRONTIER, a multidisciplinary and multi-objective optimization and design environment, written to allow coupling to almost any computer.

What is a Computer’s Role in SDR?

A software-defined radio system is a radio communication system in which certain hardware components are implemented in software. These hardware components include filters, amplifiers, modulators, and demodulators. Because these components are defined in software, you can adjust a software-defined radio system as needed without making significant hardware changes. Since computers today may contain very fast processors and high-speed interfaces, we can leverage these abilities for our software defined radio by implementing them on a computer quickly, using LabVIEW.

Driver Software

Driver software provides application software the ability to interact with a device. It simplifies communication with the device by abstracting low-level hardware commands and register-level programming. Typically, driver software exposes an application programming interface (API) that is used within a programming environment to build application software. For the USRP, NI-USRP is the hardware driver. As mentioned below, NI offers development environments that can make driver calls into NI-USRP, but other text-based environments can also access the hardware driver.

Application Software

Application software facilitates the interaction between the computer and user for acquiring, analyzing, processing, and presenting measurement data. It is either a prebuilt application with predefined functionality, or a programming environment for building applications with custom functionality. Custom applications are often used to automate multiple functions of a device, perform signalprocessing algorithms, and display custom user interfaces. The NI-USRP driver currently supports the National Instruments’ LabVIEW graphical development environment software for rapidly developing custom applications

General Information/License Agreement Terms & Conditions

- **Eligibility:** Faculty, staff, and students
- **Version:** 2014 & 2012
- **Platform:** Windows or Mac
- **Cost:** Free - for driver download only
- **Computer and Software Requirements**
  - Visit NI LabVIEW for System Requirements webpage
- **Order/Download:**
  - LabVIEW Download
- **License Restrictions:** Only those identified in the software or per the vendor
- **Notes (Instructions & Documentation):**
  - For instructions on installing LabVIEW, please visit NI's support site
- **Support:**
  - National Instruments Online Support

Project Integration

Required Equipment

Hardware

- NI USRP-2920 device
- NI USRP universal power adapter
- Gigabit (Gb) Ethernet cable
- Antenna
- Computer with a spare Gigabit (Gb) Ethernet port

Software

- NI LabVIEW 2012 or later
- NI USRP 1.1 or later
- NI Modulation Toolkit
- LabVIEW MathScriptRT Add On

Configuring Hardware

The following steps are an overview of the *NI USRP-292x Getting Started Guide*. Follow these steps after you install LabVIEW on your computer:

1. Install the NI USRP Software Suite DVD. The software suite adds the following items to your LabVIEW installation: the NI-USRP driver, LabVIEW Modulation Toolkit, LabVIEW MathScriptRT module, and LabVIEW Digital Filter Design Toolkit. For more information, refer to the Installing the Software section of this document.

2. Referring to Figure 1, attach the antenna to the front of the NI USRP-292x.

3. Connect the device directly to your computer with the enclosed Ethernet cable and plug in the power. For more information, refer to the Installing and Configuring the Hardware section of this document.

4. Change the IP address of your computer's Ethernet port to a static IP. NI recommends a static IP address of 192.168.10.1 because NI USRP-2920 has a default IP address of 192.168.10.2. For more information, refer to the Installing and Configuring the Hardware section (page 8).

5. Examples are located in LabVIEW. Navigate to the start menu and select **Start » All Programs » National Instruments » NI-USRP » Examples**. For more information, refer to the Programming the NI 292x section of this document on page 11.

6. If you have trouble with any of these steps, refer to the Appendix of this document or watch the video at ni.com/usrp/gettingstarted

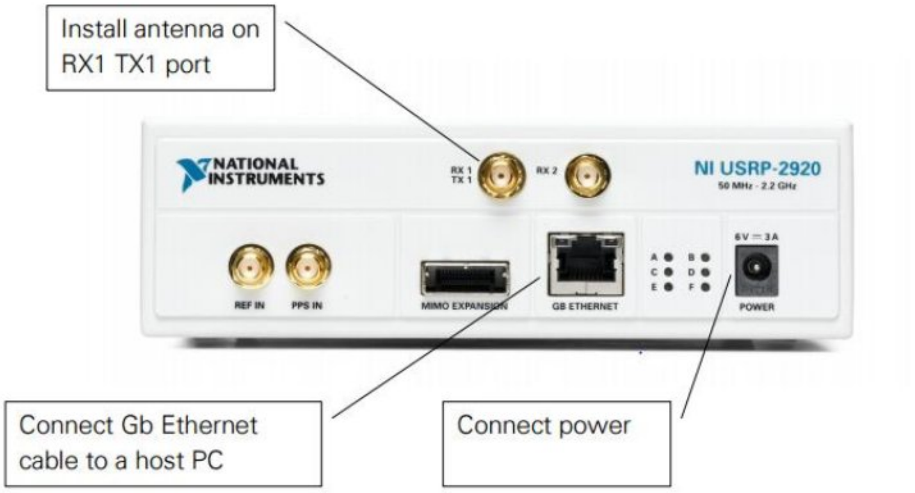


Figure 1. USRP Front Panel

Finding Hardware & Installing Software

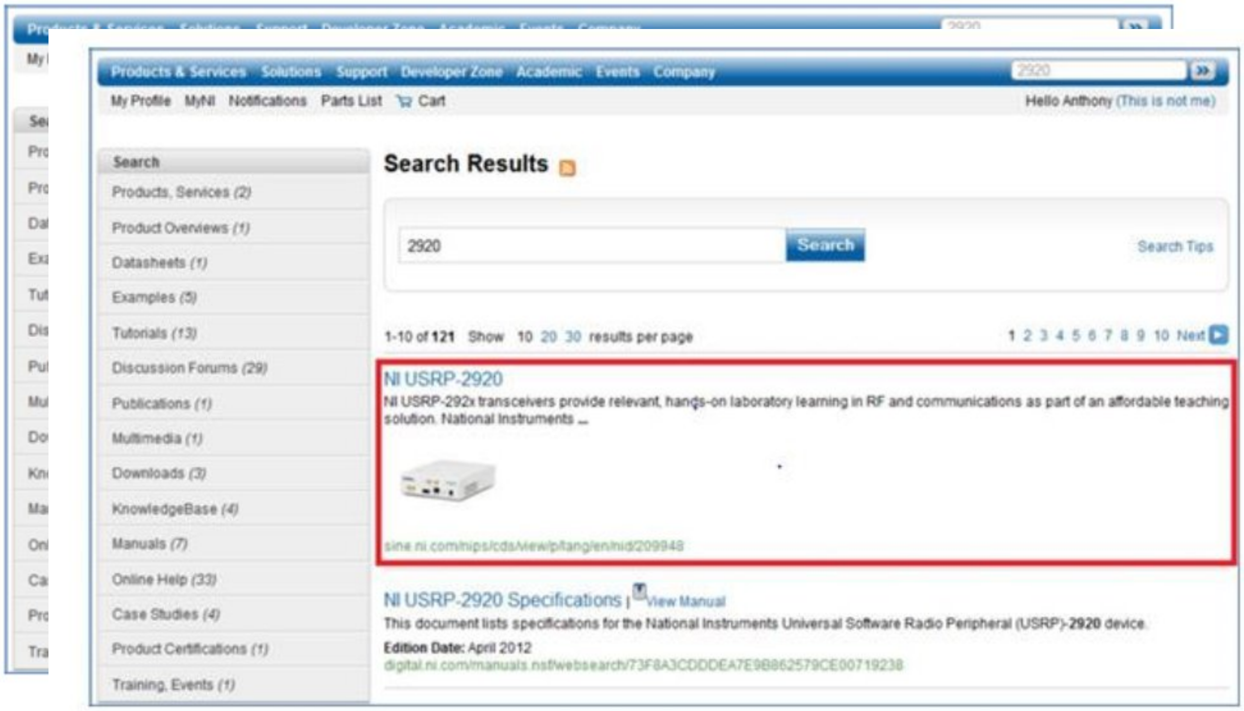
It is first important to understand the complexities and requirements of your application to ensure that your wireless signal system is the right fit.

Identifying the Right Hardware

To help you identify the right hardware for your system, first navigate to ni.com/usrp. At this portal you will find everything that you need to know about the USRP. You can use the left-hand navigation facets to begin

ed on
using
ımber
page

ware


Products & Services Solutions Support Developer Zone Academic Events Company

My Profile MyNI Notifications Parts List Cart — Hello Anthony (This is not me)

Search Results

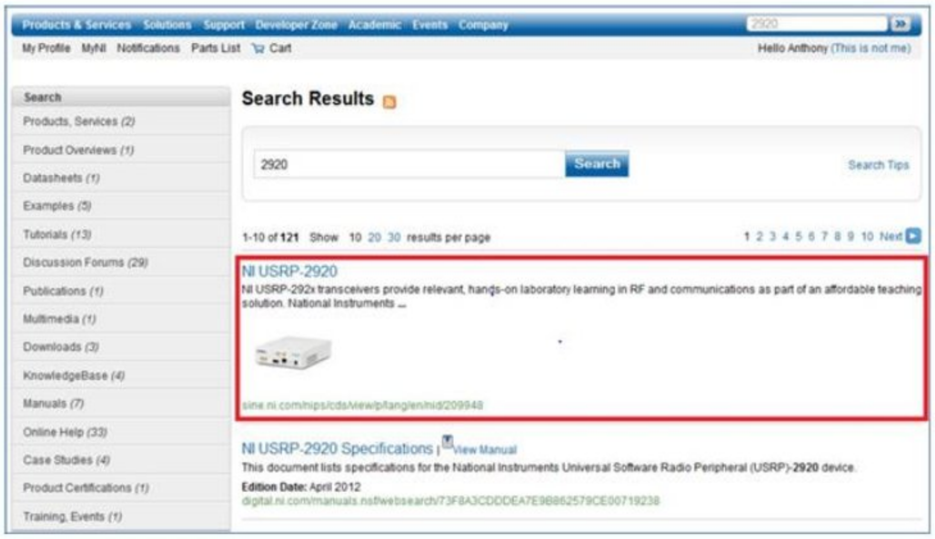
2920 Search — Search Tips

Search
Products, Services (2)
Product Overviews (1)
Datasheets (1)
Examples (5)
Tutorials (13)
Discussion Forums (29)
Publications (1)
Multimedia (1)
Downloads (3)
KnowledgeBase (4)
Manuals (7)
Online Help (33)
Case Studies (4)
Product Certifications (1)
Training, Events (1)

1-10 of 121 Show 10 20 30 results per page — 1 2 3 4 5 6 7 8 9 10 Next

NI USRP-2920
NI USRP-292x transceivers provide relevant, hands-on laboratory learning in RF and communications as part of an affordable teaching solution. National Instruments ...
sine.ni.com/nips/cds/view/p/lang/en/nid/209948

NI USRP-2920 Specifications | View Manual
This document lists specifications for the National Instruments Universal Software Radio Peripheral (USRP)-2920 device.
Edition Date: April 2012
digital.ni.com/manuals.nsf/websearch/73F8A3CDDDEA7E9B862579CE00719238


Products & Services Solutions Support Developer Zone Academic Events Company

My Profile MyNI Notifications Parts List Cart — Hello Anthony (This is not me)

Search Results

2920 Search — Search Tips

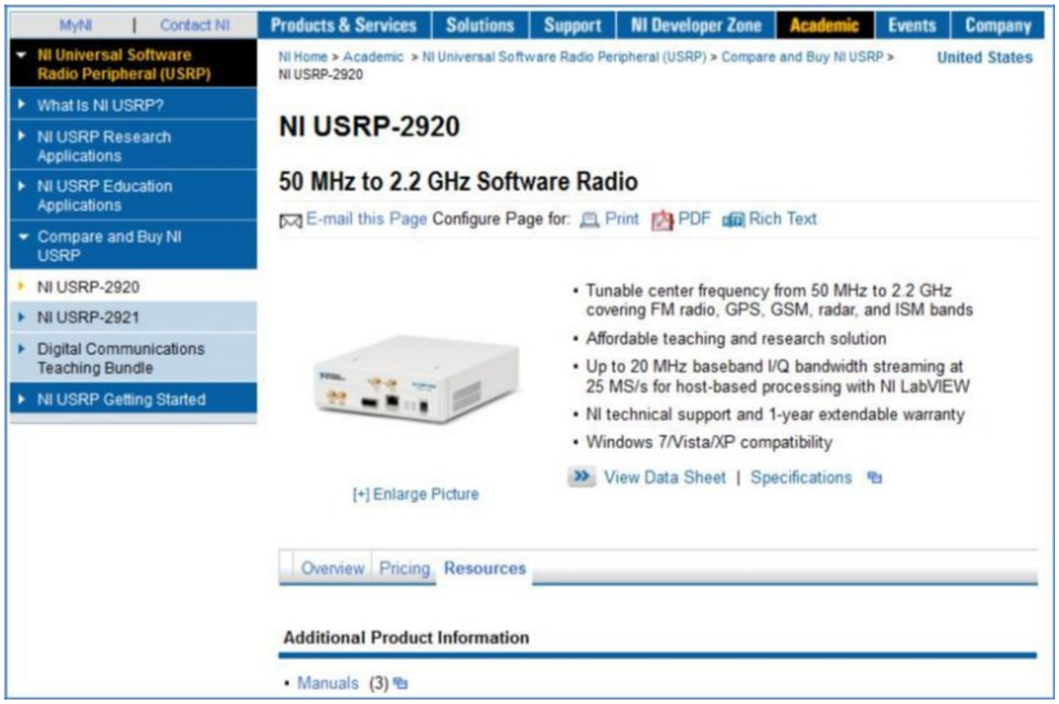
Search: Products, Services (2); Product Overviews (1); Datasheets (1); Examples (5); Tutorials (13); Discussion Forums (29); Publications (1); Multimedia (1); Downloads (3); KnowledgeBase (4); Manuals (7); Online Help (33); Case Studies (4); Product Certifications (1); Training, Events (1)

1-10 of 121 Show 10 20 30 results per page — 1 2 3 4 5 6 7 8 9 10 Next

NI USRP-2920
NI USRP-292x transceivers provide relevant, hands-on laboratory learning in RF and communications as part of an affordable teaching solution. National Instruments ...
sine.ni.com/nips/cds/view/p/lang/en/nid/209948

NI USRP-2920 Specifications | View Manual
This document lists specifications for the National Instruments Universal Software Radio Peripheral (USRP)-2920 device.
Edition Date: April 2012
digital.ni.com/manuals.nsf/websearch/73F8A3CDDDEA7E9B862579CE00719238

Once on the product page, click on the **Resources** tab to surface documents about the device; you can also access the complete **NI USRP Specifications** document for the device by clicking on the **Manuals** link.


MyNI | Contact NI
Products & Services
Solutions
Support
NI Developer Zone
Academic
Events
Company
NI Universal Software Radio Peripheral (USRP)
What Is NI USRP?
NI USRP Research Applications
NI USRP Education Applications
Compare and Buy NI USRP
NI USRP-2920
NI USRP-2921
Digital Communications Teaching Bundle
NI USRP Getting Started
NI Home > Academic > NI Universal Software Radio Peripheral (USRP) > Compare and Buy NI USRP > NI USRP-2920
United States
E-mail this Page
Configure Page for: Print PDF Rich Text
[+] Enlarge Picture
View Data Sheet | Specifications
Overview
Pricing
Resources
Manuals (3)

NI USRP-2920

50 MHz to 2.2 GHz Software Radio

- Tunable center frequency from 50 MHz to 2.2 GHz covering FM radio, GPS, GSM, radar, and ISM bands
- Affordable teaching and research solution
- Up to 20 MHz baseband I/Q bandwidth streaming at 25 MS/s for host-based processing with NI LabVIEW
- NI technical support and 1-year extendable warranty
- Windows 7/Vista/XP compatibility

Additional Product Information

<u>Purchasing Hardware</u>

To receive a quote for NI USRP Hardware, you can add the hardware to your cart at ni.com from the product page for the product. You can also call your NI Sales Engineer to help design the best system for your application.

To speak with a technical representative, navigate to ni.com and click on the **Contact Us** link on the homepage.

<u>Installing Software</u>

To acquire data from the USRP, you will need to first install a software development environment and then the hardware driver.

<u>Development Environment: NI LabVIEW</u>

The development environment facilitates the interaction between the computer and user for acquiring, processing, analyzing, and presenting measurement data. It is either a prebuilt application with predefined functionality, or a programming environment for building applications with custom functionality. Custom applications are often used to automate multiple functions of a device, perform signal-processing algorithms, and display custom user interfaces.

The primary development environment for NI-USRP is NI LabVIEW. LabVIEW is a graphical programming language that abstracts the low-level complexities of text-based programming into a visual language that scientists and engineers use worldwide to acquire, analyze, process, and present data in the same environment.

Visit ni.com/trylabview for more information and to download an evaluation.

Hardware Driver: NI-USRP

Driver software provides application software the ability to interact with a device. It simplifies communication with the device by abstracting low-level hardware commands and register-level programming. Typically, driver software exposes an application-programming interface (API) that is used within a programming environment to build application software.

The hardware driver for the USRP is NI-USRP. The USRP driver comes with example LabVIEW programs and help files to get you started.

Visit ni.com/drivers and search for ‘USRP’ to download the latest version of the driver.

Academic Site License

If you are on a university with access to an Academic Site License (ASL), then you might have free access to almost all NI software, including LabVIEW. Contact your university software administrator for more information.

NI-USRP LabVIEW Driver

When installed the NI USRP hardware driver automatically installs example programs, help files, and functions.

NI-USRP Example Programs

To locate example files for using the USRP driver in LabVIEW, navigate to the **Start Menu » All Programs » National Instruments » NI-USRP » Examples.**

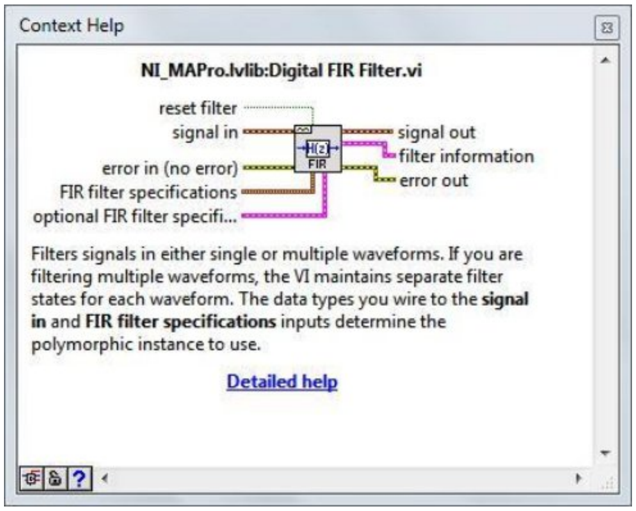
NI-USRP Help Files

There are two forms of help offered within the LabVIEW development environment: Context Help and Detailed Help.

Context Help

The context help window displays basic information about LabVIEW objects when you move the cursor over each object. To toggle the display of the context help window, select **Help » Show Context Help** or press <Ctrl-H>.

When you move the cursor over front panel and block diagram objects, the context help window displays the icon for subVIs, functions, constants, controls, and indicators, with wires attached to each terminal. When you move the cursor over dialog box options, the context help window displays descriptions of those options.



Filters signals in either single or multiple waveforms. If you are filtering multiple waveforms, the VI maintains separate filter states for each waveform. The data types you wire to the **signal in** and **FIR filter specifications** inputs determine the polymorphic instance to use.

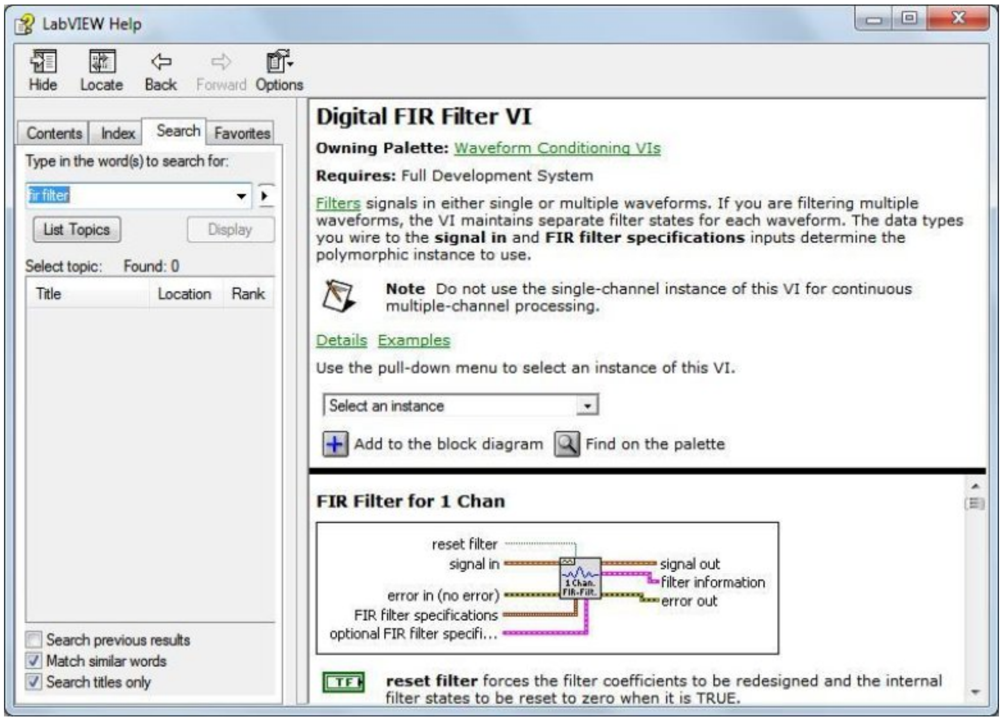
Figure 7. LabVIEW Context Help

If a corresponding detailed help topic exists for an object that the context help window describes, a blue **detailed help** link appears in the context help window (Figure 7). Click the link or the button to display the LabVIEW help for more information about the object.

Detailed Help (LabVIEW Help)

The LabVIEW help is the best source of detailed information about specific features and functions in LabVIEW. Detailed help entries break down topics into a concepts section with detailed descriptions and a how-to section with step-by-step instructions for using LabVIEW functions.

You can access the LabVIEW Help by selecting **Help » Search the LabVIEW Help**, or clicking the blue **Detailed help** link in the context help window. You also can right-click an object and select **Help** from the shortcut menu.



Type in the word(s) to search for: fir filter

Select topic: Found: 0

Title	Location	Rank

☐ Search previous results
☑ Match similar words
☑ Search titles only

Digital FIR Filter VI

Owning Palette: Waveform Conditioning VIs

Requires: Full Development System

Filters signals in either single or multiple waveforms. If you are filtering multiple waveforms, the VI maintains separate filter states for each waveform. The data types you wire to the **signal in** and **FIR filter specifications** inputs determine the polymorphic instance to use.

Note Do not use the single-channel instance of this VI for continuous multiple-channel processing.

Details Examples

Use the pull-down menu to select an instance of this VI.

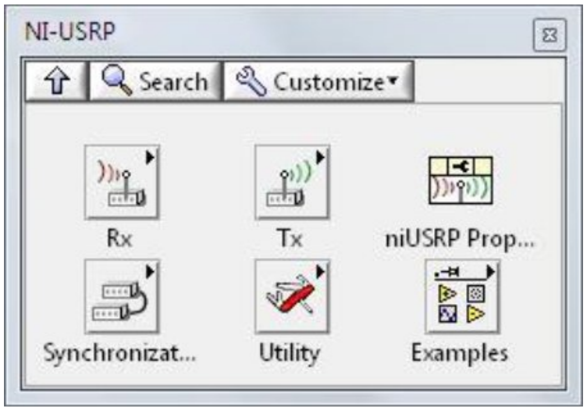
Select an instance

FIR Filter for 1 Chan

reset filter forces the filter coefficients to be redesigned and the internal filter states to be reset to zero when it is TRUE.

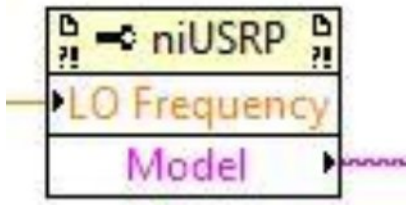
NI-USRP Functions Palette

To access the NI-USRP functions in LabVIEW, navigate to the block diagram and right-click empty white space to bring up the functions palette. Then navigate to **Instrument Drivers » NI-USRP**. The functions will appear similar to the palette below. Drag and drop a function onto the block diagram to begin programming.



Ni USRP Property Node

Use the niUSRP properties to access advanced configuration options for NI-USRP driver applications.



The Eight Most-Used NI-USRP Functions

The following section outlines the eight most-used USRP functions to help get you started with your experiments. They have been grouped in categories by their functionality. These categories are: Configure, Read/Write, and Close. These categories are included in most data acquisition programs and are important programming models to consider when creating a new LabVIEW Virtual Instrument (VI).

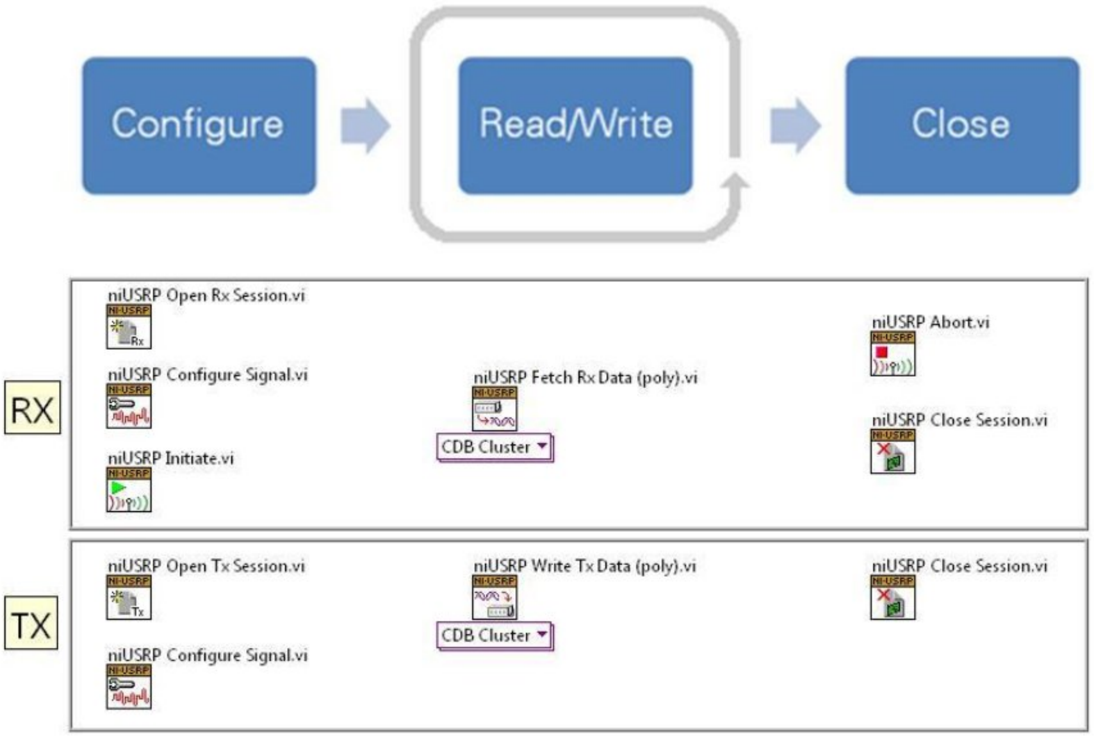

Configure
Read/Write
Close
RX
niUSRP Open Rx Session.vi
niUSRP Configure Signal.vi
niUSRP Initiate.vi
niUSRP Fetch Rx Data (poly).vi
CDB Cluster
niUSRP Abort.vi
niUSRP Close Session.vi
TX
niUSRP Open Tx Session.vi
niUSRP Configure Signal.vi
niUSRP Write Tx Data (poly).vi
CDB Cluster
niUSRP Close Session.vi

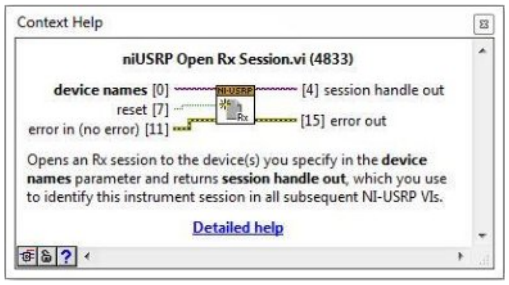
Figure 11. The Eight Most-Used NI-USRP Functions

Configure Functions

Ni-USRP Open Rx Session

The niUSRP Open Rx Session VI is the first VI that is used to create a software session with the USRP for receiving an RF signal. A session is necessary to send configuration data and retrieve IQ data from the USRP.

An Rx session can only be used with Rx functions.



Context Help

niUSRP Open Rx Session.vi (4833)

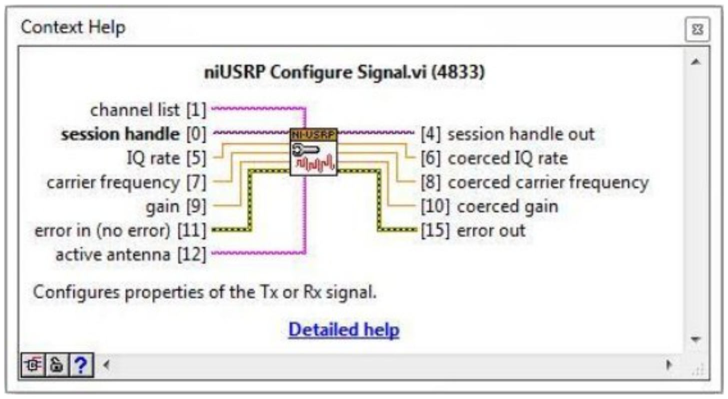
Opens an Rx session to the device(s) you specify in the **device names** parameter and returns **session handle out**, which you use to identify this instrument session in all subsequent NI-USRP VIs.

Detailed help

Ni USRP Configure Signal

The niUSRP Configure Signal VI can be used with a receive (Rx) or a transmit (Tx) session. It sets the IQ rate, carrier frequency, gain, and active antenna. For multiple USRP configurations

the channel list specifies a specific USRP. Not all IQ rates, frequencies and gains are valid. Always read the coerced values to see if the requested and actual (coerced) values are different.


Context Help

niUSRP Configure Signal.vi (4833)

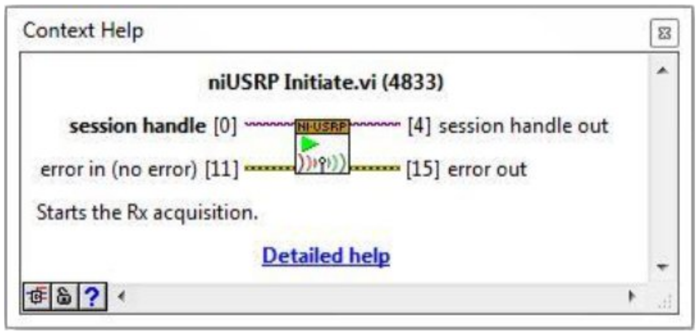
Configures properties of the Tx or Rx signal.

Detailed help

Ni USRP Initiate

The niUSRP Initiate VI starts the receive session and tells the USRP that all configuration is complete and that the USRP should begin to capture IQ data (samples).

This VI can only be used with an Rx session.


Context Help

niUSRP Initiate.vi (4833)

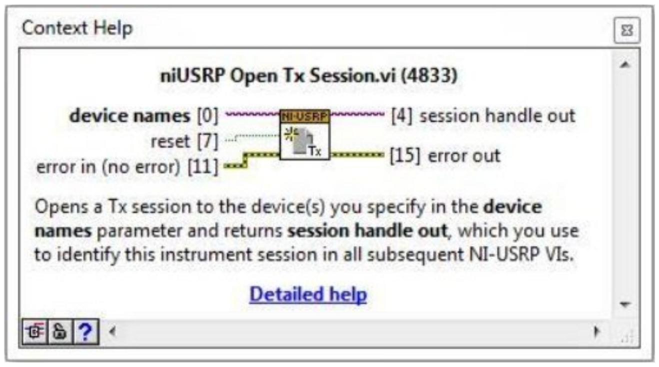
Starts the Rx acquisition.

Detailed help

Ni USRP Open Tx Session

The niUSRP Open Tx Session VI is the first VI that is used to create a connection to the USRP for transmitting an RF signal. A session is necessary to send configuration data and send IQ

A Tx session can only be used with Tx functions.



Context Help

niUSRP Open Tx Session.vi (4833)

Opens a Tx session to the device(s) you specify in the **device names** parameter and returns **session handle out**, which you use to identify this instrument session in all subsequent NI-USRP VIs.

Detailed help

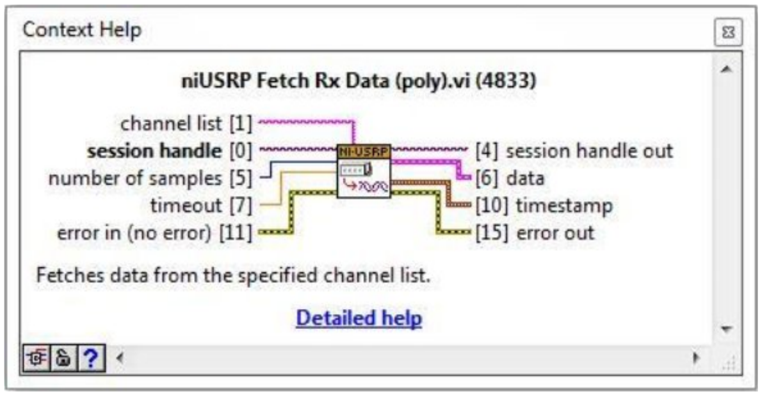
Read/Write Functions

Ni USRP Fetch Rx Data (Polymorphic)

The niUSRP Fetch Rx Data VI allows you to retrieve IQ data from a USRP that has an Rx session created with the niUSRP Open Rx Session VI. This data can then be graphed in time domain, or digitally processed for analysis.

This VI is polymorphic, meaning that there are several versions (instances) of the VI available to choose from depending on the **data type** you wish to work with.

This VI can only be used with an Rx session.



Context Help

niUSRP Fetch Rx Data (poly).vi (4833)

Fetches data from the specified channel list.

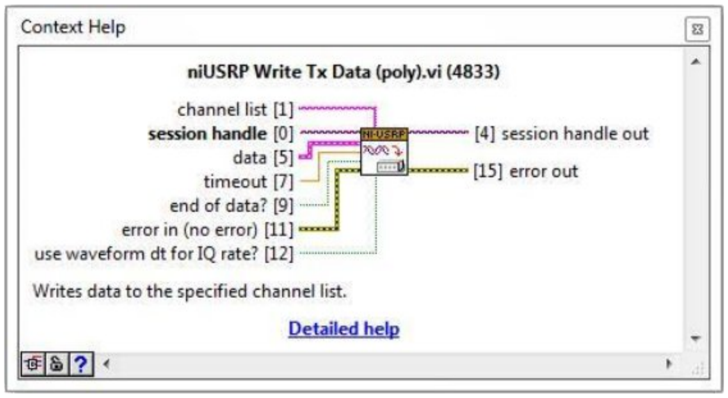
Detailed help

Ni USRP Write Tx Data (Polymorphic)

The niUSRP Write Tx Data VI allows you to send IQ data to the USRP so that it may transmit that data at the carrier frequency specified by the niUSRP Configure Signal VI.

This VI is polymorphic, meaning that there are several versions (instances) of the VI available to choose from depending on the **data type** you wish to work with.

This VI can only be used with a Tx session.



niUSRP Write Tx Data (poly).vi (4833)

Writes data to the specified channel list.

NI-USRP Read and Write Data Types

There are several instances of Write Tx Data and Fetch Rx Data VIs to choose from for your convenience. The table below represents the options available instances.

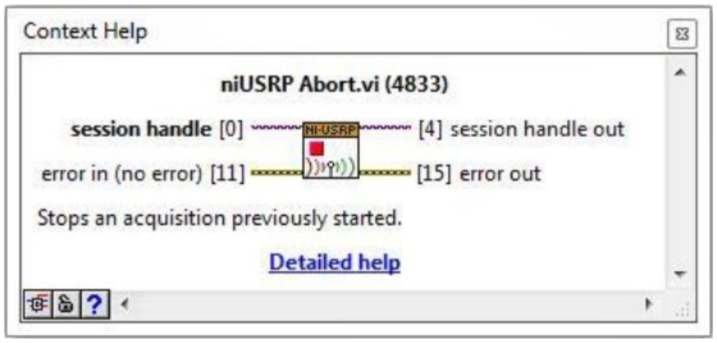
Polymorphic Type	Description
Complex Double Cluster	Fetches a cluster of complex, double-precision floating-point data from the specified channel. Modulation Toolkit VIs use the complex, double-precision floating-point cluster data type. Use this VI in applications that use Modulation Toolkit VIs.
Complex Double Waveform Data	Fetches complex, double-precision floating-point data in a waveform data type from the specified channel.
Complex Double	Fetches complex, double-precision floating-point data from the specified channel.
16-bit Integer	Fetches complex, 16-bit signed integer data from the specified channel. To use this VI, you must set the Host Data Type property to I16.
2D Array Complex Double	Fetches complex, double-precision floating-point data from multiple channels.
2D Array 16-bit Integer	Fetches complex, 16-bit signed integer data from multiple channels. To use this VI, you must set the Host Data Type property to I16.

Close Functions

Ni USRP Abort

The niUSRP Abort VI tells the USRP to stop an acquisition in progress. This VI allows you to change configuration settings without completely closing the session and creating a new session.

This VI can only be used with an Rx session.


Context Help

niUSRP Abort.vi (4833)

session handle [0] → [4] session handle out

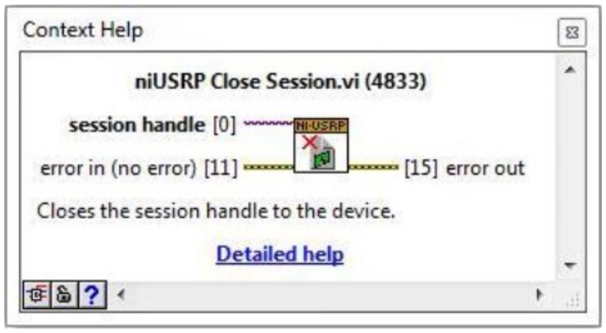
error in (no error) [11] → [15] error out

Stops an acquisition previously started.

Detailed help

Ni USRP Close Session

The niUSRP Close Session VI closes the current Rx or Tx session and releases the memory in use by that session. After calling this VI you can no longer transmit to or receive data from the USRP until you re-open a new session.


Context Help

niUSRP Close Session.vi (4833)

session handle [0]

error in (no error) [11] → [15] error out

Closes the session handle to the device.

Detailed help

USRP Receive and Transmit Examples

The following examples are the skeleton programs for transmitting and receiving. Each example will have a few example applications, along with example LabVIEW block diagram code to help get you started. These examples can be found at **Start » All Program » National Instruments » NI USRP » Examples**

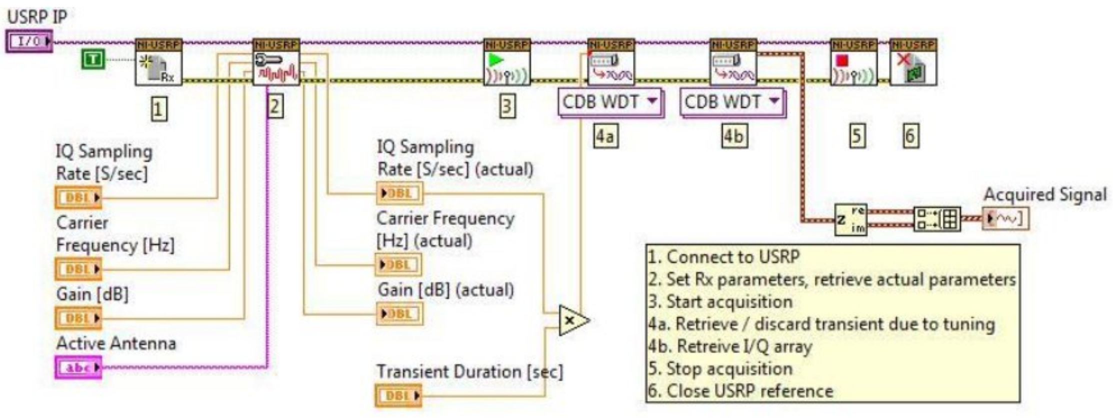
Single Channel, Finite

Applications: Burst data transmits, frequency hopping, large frequency sweeps
Example VI Name: niUSRP EX One Shot Rx.vi

The following table lists out the VIs that you will use for this type of application. Note that some VIs are required, others can be added for more advanced functionality and some are not applicable.

NI-USRP VI	Acquire	Transmit
Open Session	□	□
Configure Signal	□	□
Initiate	□	
Fetch/Write Data	□	□
While Loop		
USRP Property Node	□	□
Abort	□	
Close Session	□	□

The single channel finite example can be used as a starting point for a simple transmitter that transmits bursts of data. A while loop is not needed if only a single burst of data needs to be transmitted. Using a while loop and the abort and initiate VIs we can change the frequency of the carrier frequency on the fly without closing and reopening Rx sessions (which takes extra time). For finite Tx sessions, a boolean indicator called **End of Data?** is set to true to let the USRP turn off the transmitter after the last sample has been transmitted.



1. Connect to USRP
2. Set Rx parameters, retrieve actual parameters
3. Start acquisition
4a. Retrieve / discard transient due to tuning
4b. Retreive I/Q array
5. Stop acquisition
6. Close USRP reference

Figure 20. Example block diagram code for single USRP (channel) finite acquisition

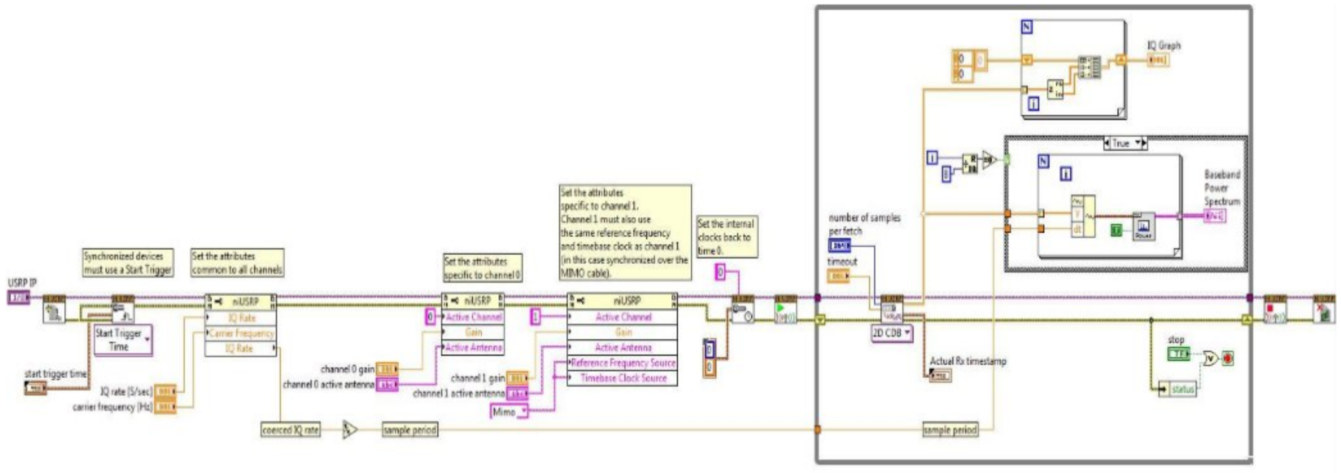
Multiple Channels, Continuous

Application: Broadband spectrum monitoring, MIMO transmission and reception **Example VI Name:** niUSRP EX Rx Multiple Synchronized Inputs.vi

The following table lists out the VIs that you will use for this type of application. Note that some VIs are required, others can be added for more advanced functionality and some are not applicable.

NI-USRP VI	Acquire	Transmit
Open Session	☐	☐
Configure Signal	☐	☐
Initiate	☐	
Fetch/Write Data	☐	☐
While Loop	☐	☐
USRP Property Node	☐	☐
Abort	☐	
Close Session	☐	☐

The multiple channel continuous example is an extension of the continuous example that is set up for multiple USRPs (channels). Multiple USRPs allow you to obtain a larger system bandwidth by using the individual USRPs at separate frequencies. To configure a full MIMO system, you invoke a LabVIEW property node and use the MIMO cable or synchronize to an external clock source.



Synchronized devices must use a Start Trigger

Set the attributes common to all channels

Set the attributes specific to channel 0

Set the attributes specific to channel 1. Channel 1 must also use the same reference frequency and timebase clock as channel 1 (in this case synchronized over the MIMO cable).

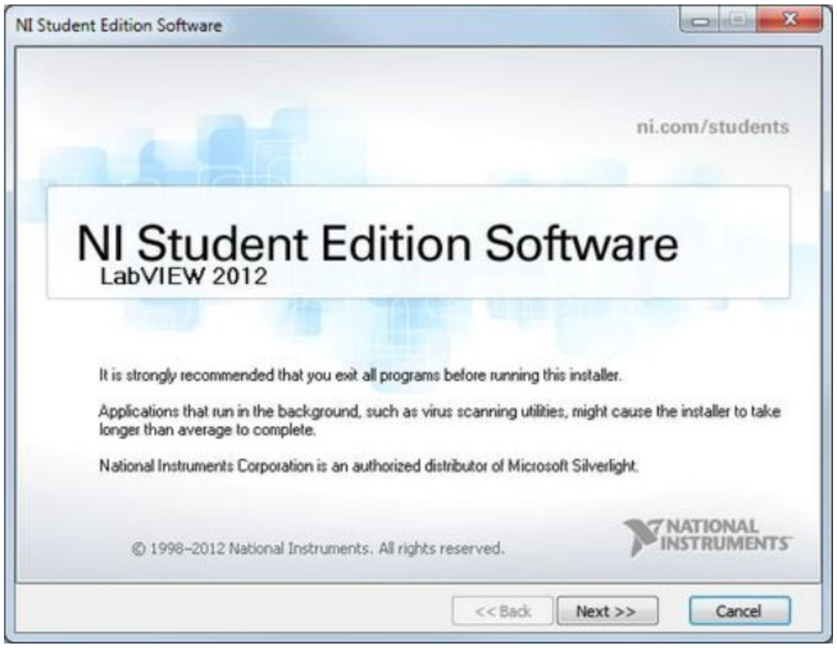
Set the internal clocks back to time 0.

Figure 22. Example block diagram code for multiple channel continuous acquisitio

Project Simulation

Installing the Requisites

- **Install NI-software edition**



NI Student Edition Software

ni.com/students

NI Student Edition Software
LabVIEW 2012

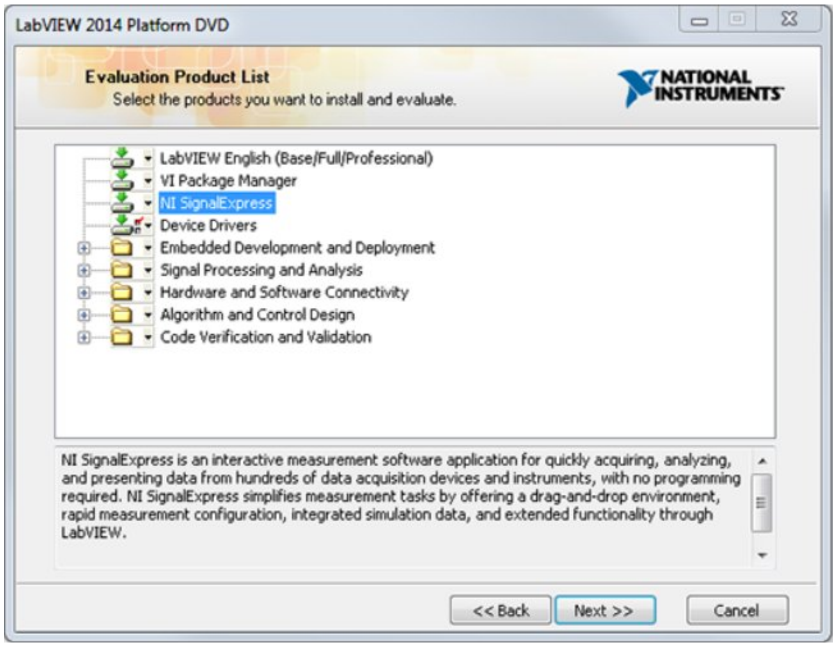
It is strongly recommended that you exit all programs before running this installer.

Applications that run in the background, such as virus scanning utilities, might cause the installer to take longer than average to complete.

National Instruments Corporation is an authorized distributor of Microsoft Silverlight.

© 1998–2012 National Instruments. All rights reserved.

- **Selecting Evaluation Products**



LabVIEW 2014 Platform DVD

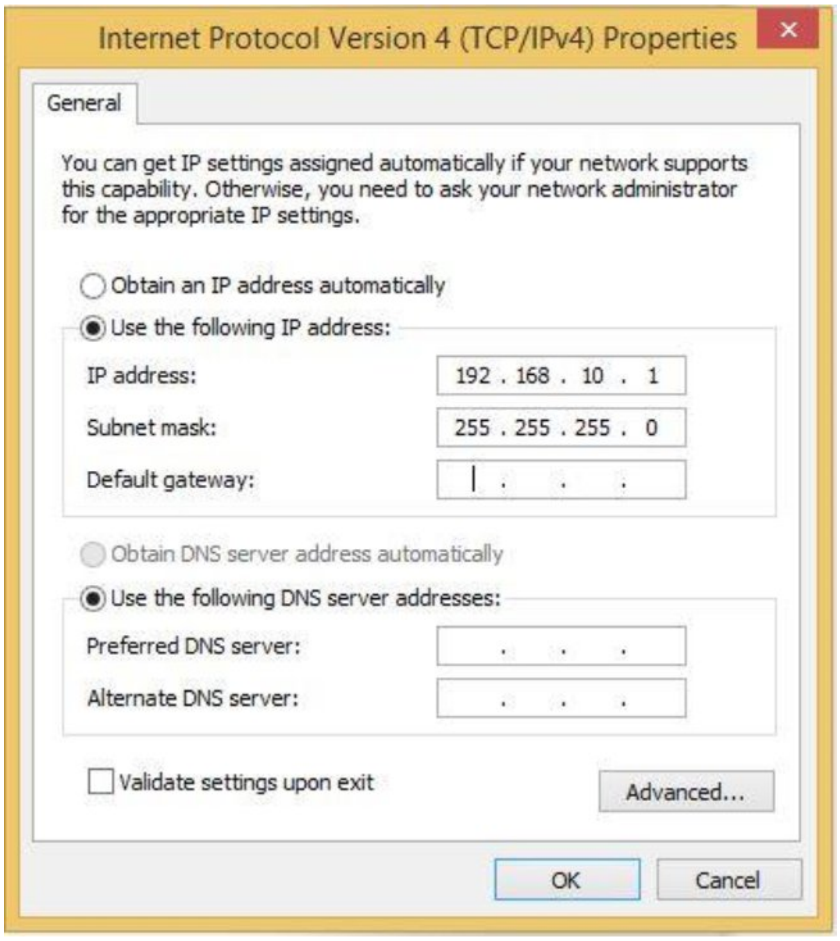
Evaluation Product List
Select the products you want to install and evaluate.

- LabVIEW English (Base/Full/Professional)
- VI Package Manager
- NI SignalExpress
- Device Drivers
- Embedded Development and Deployment
- Signal Processing and Analysis
- Hardware and Software Connectivity
- Algorithm and Control Design
- Code Verification and Validation

NI SignalExpress is an interactive measurement software application for quickly acquiring, analyzing, and presenting data from hundreds of data acquisition devices and instruments, with no programming required. NI SignalExpress simplifies measurement tasks by offering a drag-and-drop environment, rapid measurement configuration, integrated simulation data, and extended functionality through LabVIEW.

Checking availability of Usrp to PC

- For USRP connectivity provide the require IP-Addresses at the Internet Protocol Version 4 properties:



Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

( ) Obtain an IP address automatically
(●) Use the following IP address:

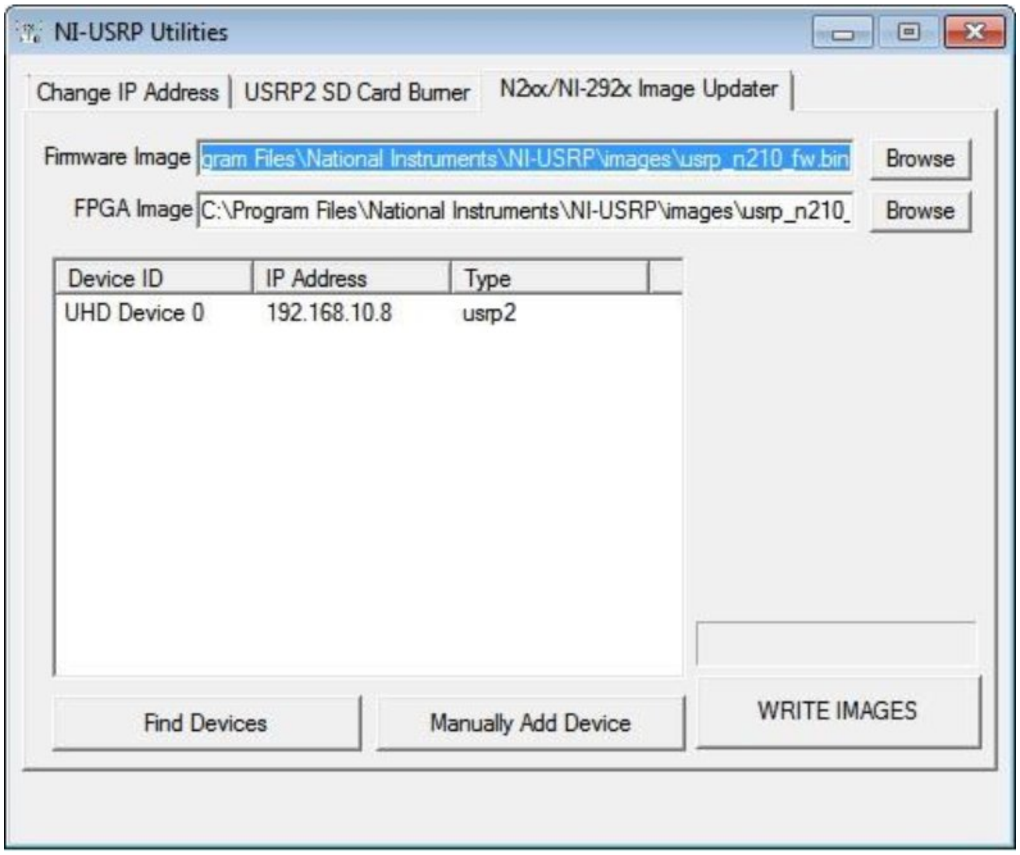
IP address:	192 . 168 . 10 . 1
Subnet mask:	255 . 255 . 255 . 0
Default gateway:	. . .

( ) Obtain DNS server address automatically
(●) Use the following DNS server addresses:

Preferred DNS server:	. . .
Alternate DNS server:	. . .

☐ Validate settings upon exit

Testing the SDR platform



NI-USRP Utilities

Change IP Address | USRP2 SD Card Burner | N2xx/NI-292x Image Updater

Firmware Image: gram Files\National Instruments\NI-USRP\images\usrp_n210_fw.bin

FPGA Image: C:\Program Files\National Instruments\NI-USRP\images\usrp_n210_

Device ID	IP Address	Type
UHD Device 0	192.168.10.8	usrp2

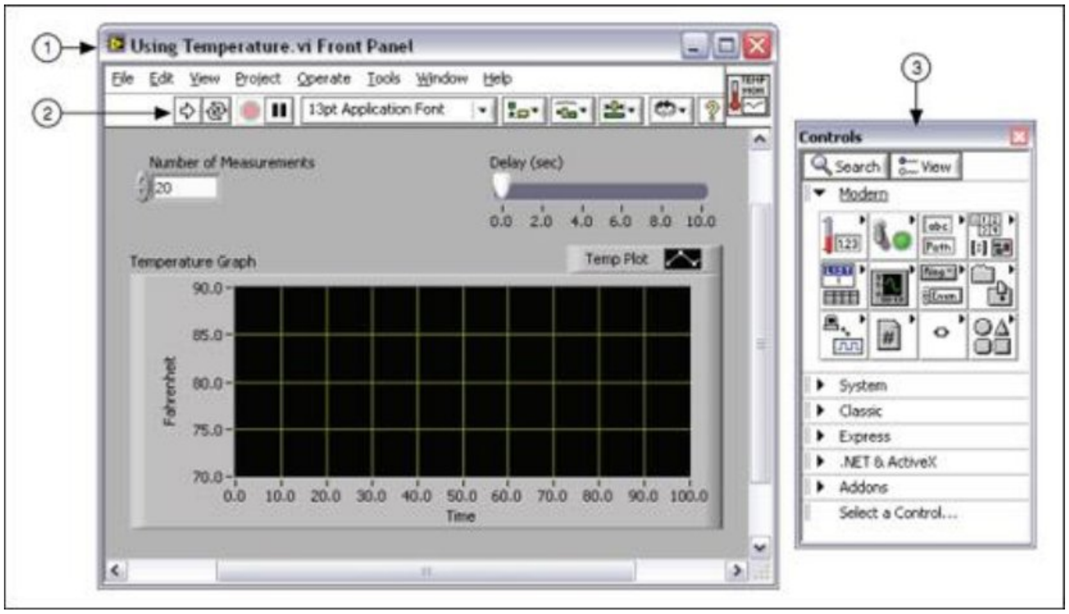
Bocks in LABVIEW

LabVIEW programs are called virtual instruments, or VIs, because their appearance and operation often imitate physical instruments, such as oscilloscopes and multimeters. LabVIEW contains a comprehensive set of tools for acquiring, analyzing, displaying, and storing data, as well as tools to help you troubleshoot the code you write.

When you create a new VI, you see two windows: the front panel window and the block diagram.

Front Panel

When you open a new or existing VI, the front panel window of the VI appears. The front panel window is the user interface for the VI.**Figure 1** shows an example of a front panel window.



Controls Palette

The Controls palette contains the controls and indicators you use to create the front panel. You access the Controls palette from the front panel window by selecting View»Controls Palette or by right clicking on any empty space in the front panel window. The Controls palette is broken into various categories; you can expose some or all of these categories to suit your needs. **Figure 2** shows a Controls palette with all of the categories exposed and the Modern category expanded.

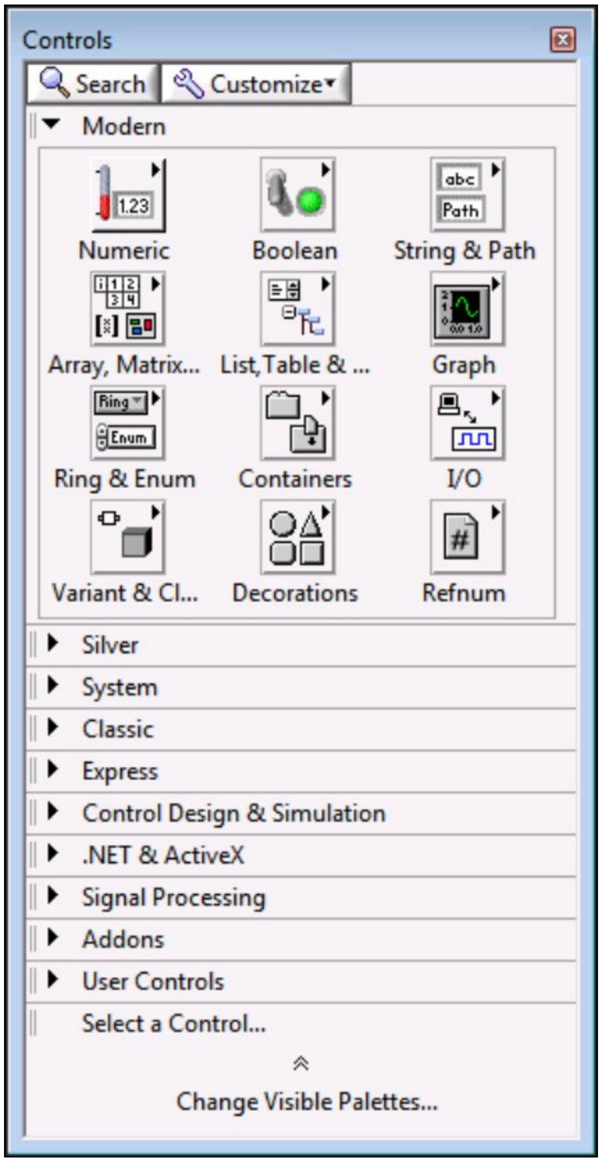
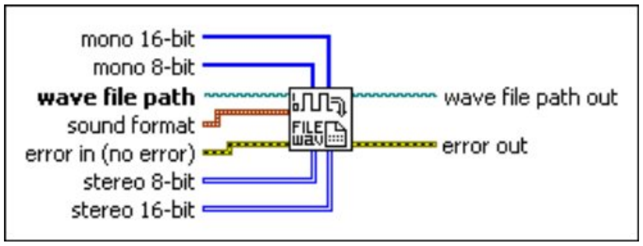


Figure 2. Controls Palette

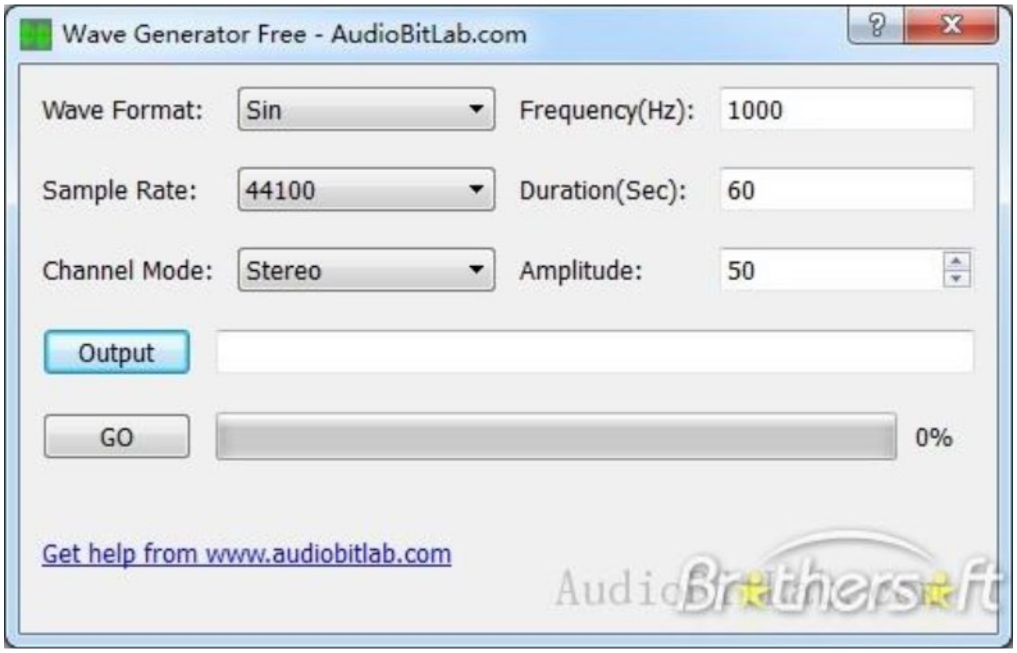
To view or hide categories (subpalettes), click the Customize button and select Change Visible Palettes.

Wave-File generation

Waveform Audio **File Format** (**WAVE**, or more commonly known as **WAV** due to its filename **extension**) (rarely, Audio for Windows) is a Microsoft and IBM audio **file format** standard for storing an audio bitstream on PCs

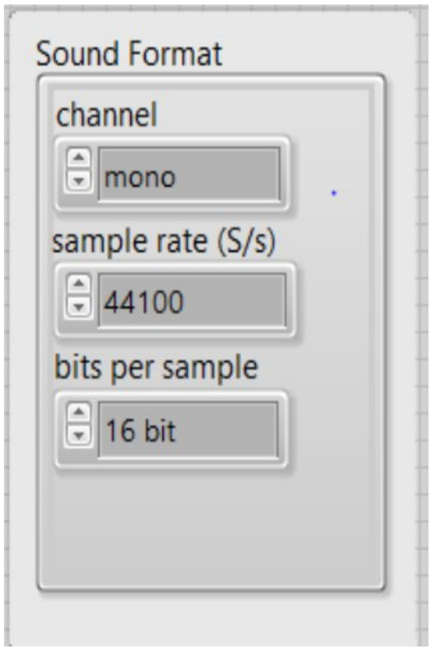


- Any recorded voice or file transfer should be initiated in the wave-file format.
- Required sample rate is set according to your program defined.


Wave Generator Free - AudioBitLab.com

Wave Format: Sin
Frequency(Hz): 1000
Sample Rate: 44100
Duration(Sec): 60
Channel Mode: Stereo
Amplitude: 50

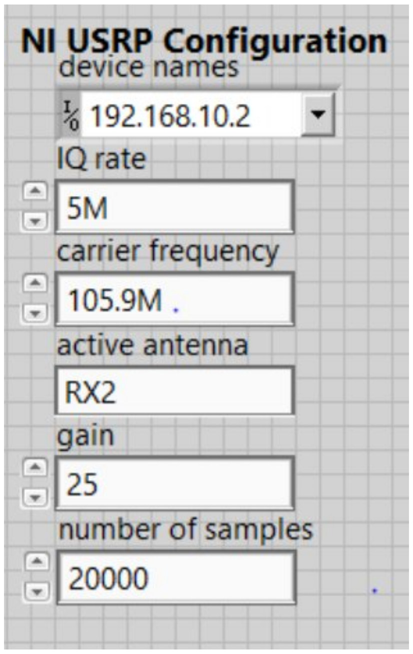
TRASMISSION PARAMETERS


Sound Format

channel: mono

sample rate (S/s): 44100

bits per sample: 16 bit


NI USRP Configuration

device names: 192.168.10.2

IQ rate: 5M

carrier frequency: 105.9M

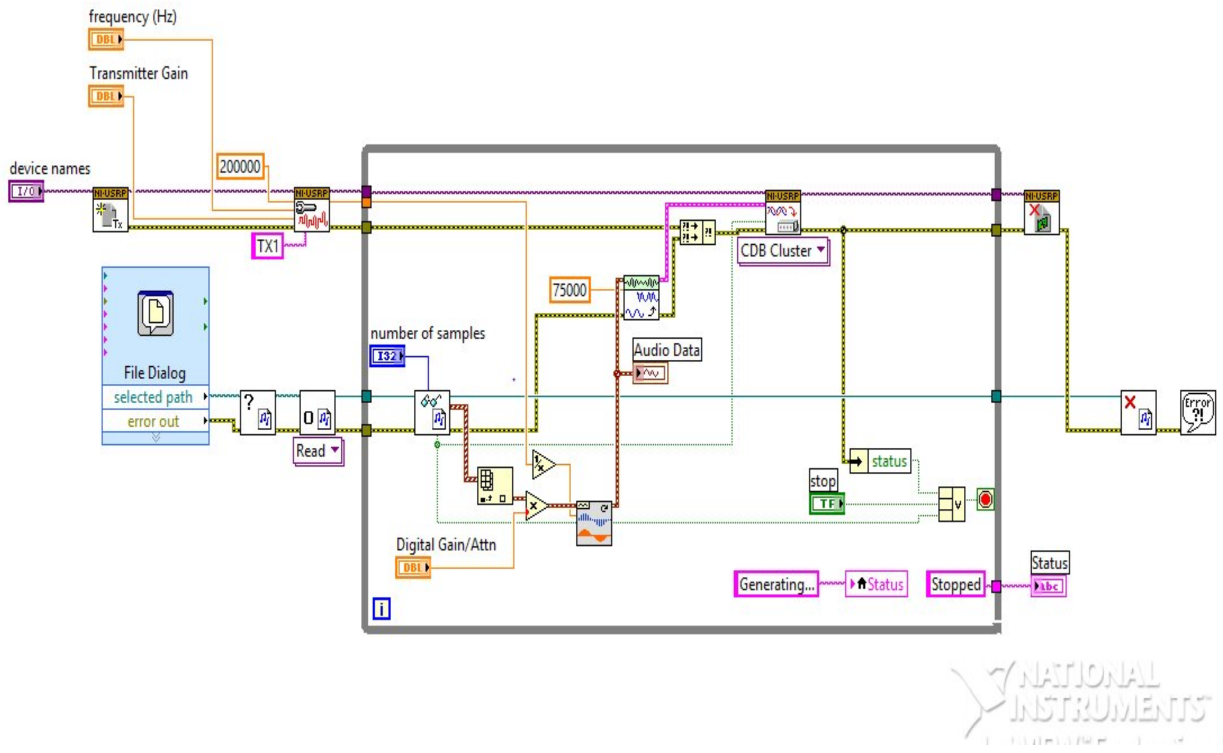
active antenna: RX2

gain: 25

number of samples: 20000

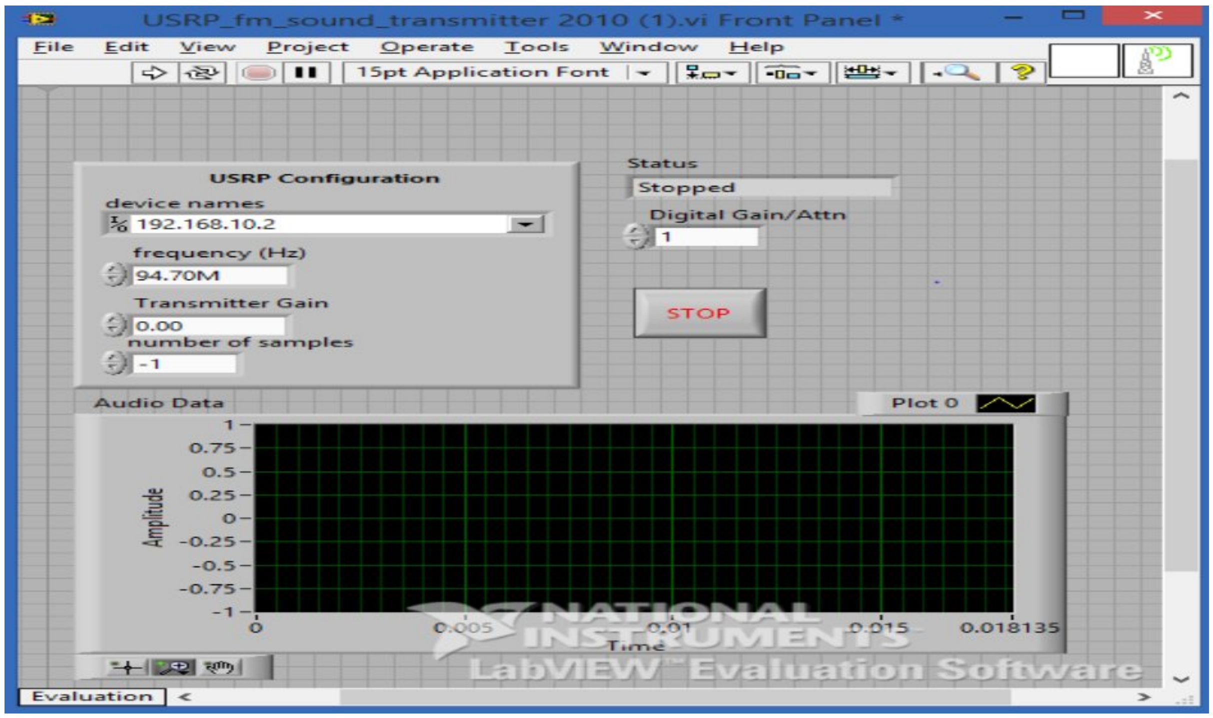
Audio Transmission

Block Diagram Configuration



F

ront Panel Depiction


USRP_fm_sound_transmitter 2010 (1).vi Front Panel *
File Edit View Project Operate Tools Window Help
15pt Application Font
STOP
Plot 0
Amplitude
Time
Evaluation
NATIONAL INSTRUMENTS LabVIEW Evaluation Software

USRP Configuration

device names: 192.168.10.2

frequency (Hz): 94.70M

Transmitter Gain: 0.00

number of samples: -1

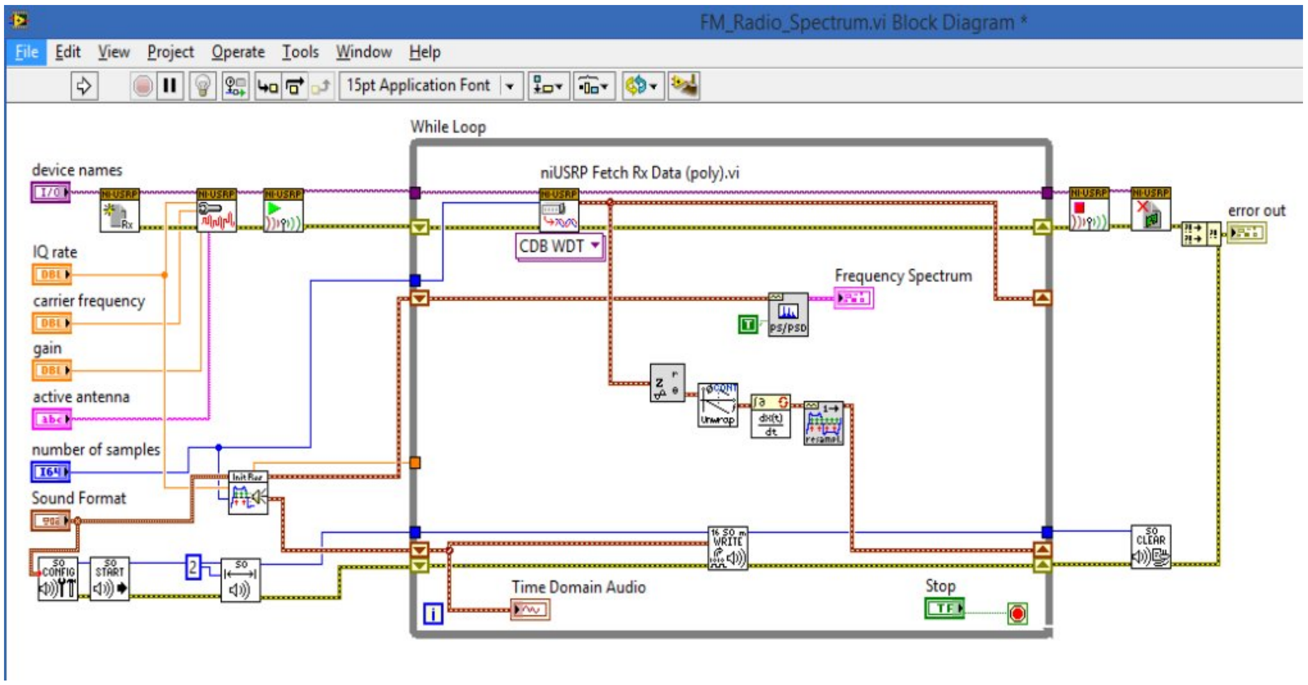
Status: Stopped

Digital Gain/Attn: 1

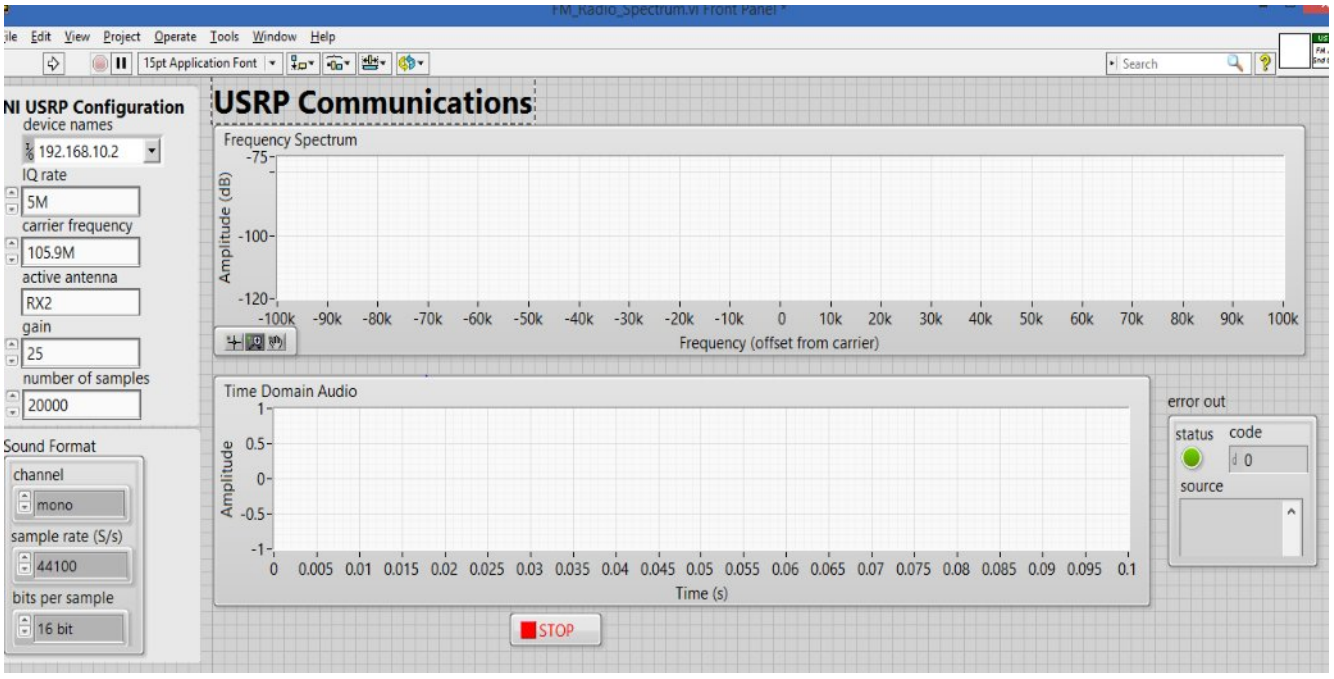
Audio Data

Audio Reception

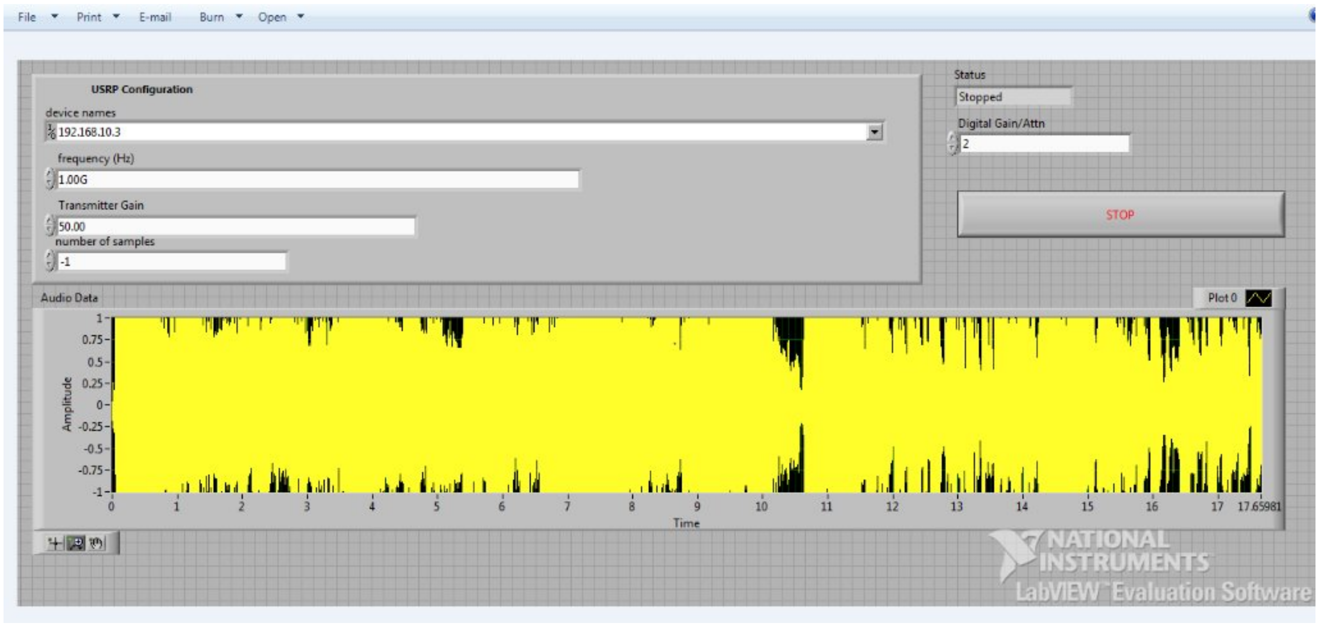
Block Diagram Configuration



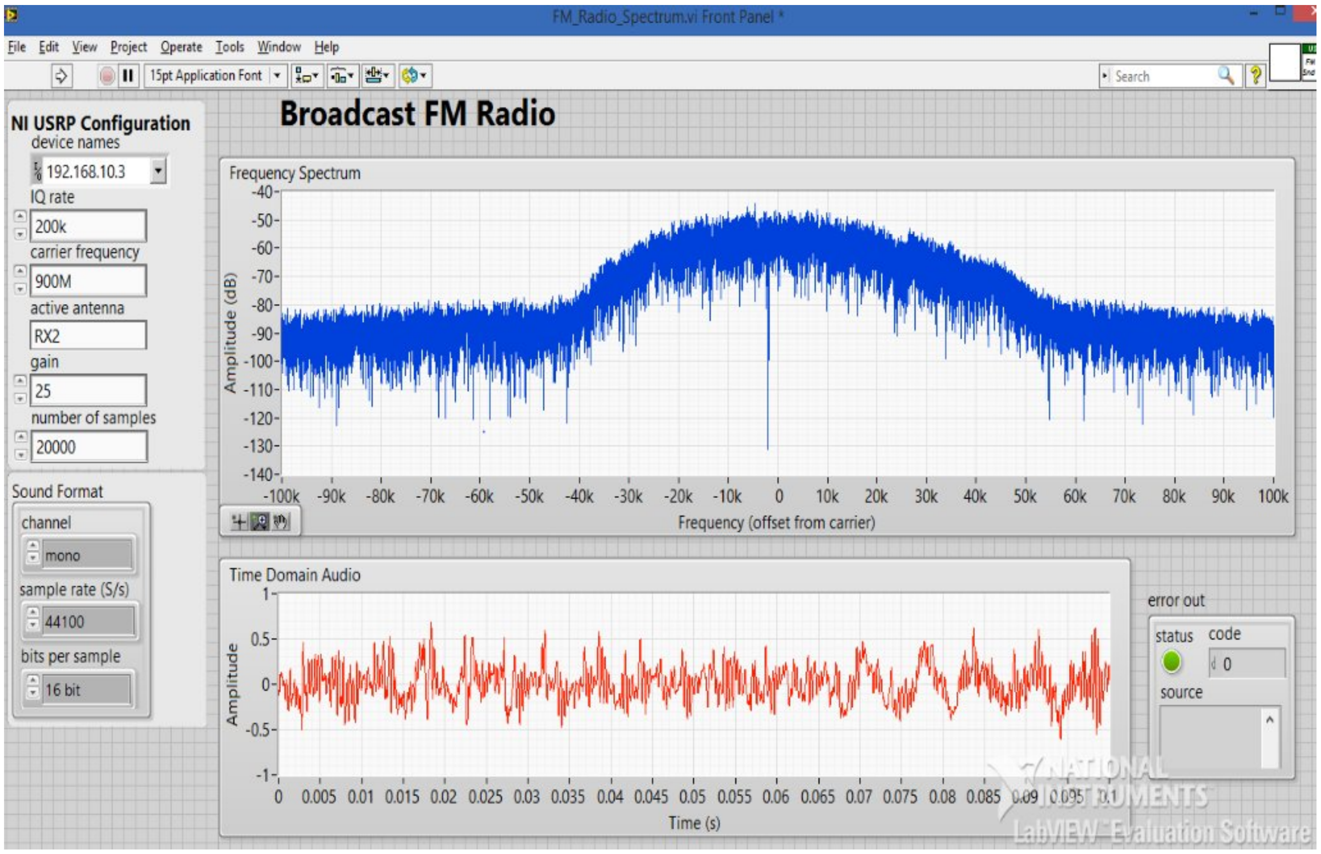
Front Panel Depiction



Transmission Simuation Results



Reception Simulation Results



Glossary A: RF & Communication Reference

Symbols

A

Amplitude Modulation (AM)	A process that varies the amplitude of a radio frequency (RF) carrier signal according to the amplitude of the message signal.
Amplitude-Shift Keying (ASK)	Refers to a type of amplitude modulation, which assigns bit values to discrete amplitude levels. The carrier signal is then modulated among the members of a set of discrete values to transmit information.
analog signal	An **analog signal** is any continuous signal for which the time varying feature (variable) of the signal is a representation of some other time varying quantity.[3]
ADC	**Analog-to-Digital Converter**—A hardware component that converts analog voltages to digitized values. An ADC can convert an analog signal to a digital signal representing equivalent information.

B

bandwidth	The measure of a circuit or transmission channel to pass a signal without significant attenuation over a range of frequencies. Bandwidth can also refer to the information rate (in bits per second) that can pass through a circuit or transmission channel.
baseband signal	The baseband is the range in the frequency spectrum occupied by the unmodulated message signal. Both the message signal and the downconverted complex I/Q signal are referred to as baseband signals.
binary signal	A signal that carries information by varying between two possible values, corresponding to 0 and 1 in the binary system.

C

carrier	The signal that carries the information encoded or modulated on it. Typically, the carrier is a fixed frequency sine wave, which can be amplitude-, phase-, or frequency-modulated.

carrier frequency	The frequency of the carrier signal that is a sinusoidal signal upon which the desired signal to be transmitted is modulated. The sinusoidal signal "carries" the modulation.
Carson's rule	Defines the approximate modulation bandwidth required for a carrier signal that is frequency-modulated by a spectrum of frequencies rather than a single frequency. The Carson bandwidth rule is expressed by the relation CBR = $2(\Delta \boldsymbol{f} + \boldsymbol{f}_m)$ where CBR is the bandwidth requirement, $\boldsymbol{f}$ is the carrier peak deviation frequency, and $\boldsymbol{f}_m$ is the highest modulating frequency.
CCDF	The **Complementary Cumulative Distribution Function** (CCDF) is a statistical characterization of the time-domain waveform that completely describes the power characteristics of a signal.
center frequency	The middle frequency of the channel bandwidth. In frequency modulation, the center frequency is equal to the *rest frequency*—specifically, the frequency of the unmodulated carrier wave.
component (I/Q)	The real (I—in-phase) and imaginary (Q—quadrature) parts of a complex number are referred to as *components*. The Modulation Toolkit VIs can use complex components to describe signal properties. Use the NI-USRP functions Write Tx Data VI and NI-USRP Fetch Data VI in applications that use Modulation Toolkit VIs, because they also use complex components.
	For example, you can represent a two-dimensional vector of length S by its components $S = A + iB$, where A and B are the vector x- and y-components. The real part of the vector corresponds to the x-component (A), while the imaginary part corresponds to the y-component (B)
D	
DC offset	A complex signal impairment that shifts the locus of ideal symbol coordinates off-center in the I/Q plane. A DC offset can be added to the baseband I component, the Q component, or both. The DC offset can be either positive or negative, with the sign indicating direction of the shift. DC offset is expressed as a percentage of full scale, where "full scale" is the amplitude of the baseband QM waveform.
DAC	Digital-to-analog converter—An electronic device, often an integrated circuit, that converts a digital number into a corresponding analog voltage or current.

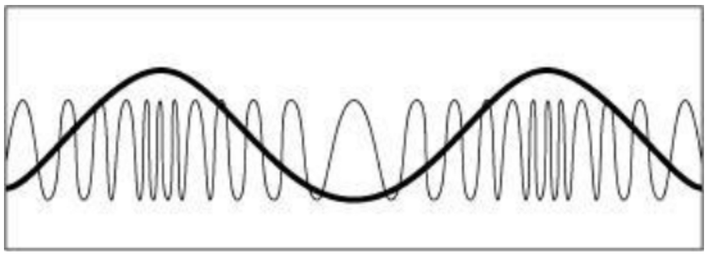
Decoding	Data decoding involves removing redundant bits from the sequence and correcting for any errors that might have happened during transmission. The signal decoding process is usually more complicated than the encoding process and can be very computationally intensive.
demodulation	Describes the recovery from a modulated wave of a signal having the same characteristics as the original message signal. The down converted signal undergoes a demodulation process. This step is the opposite of modulation and refers to the process required to extract the original information signal from the modulated signal
dB, decibel	The unit for expressing a logarithmic measure of the ratio of two signal levels: dB = 20 $\log_{10}$(V1/V2), for signals in volts.
downconverter	A signal conditioning device that converts a specific band of high-frequency (RF) signals to lower, more manageable intermediate frequencies (IF) that can be digitized.
Down conversion	The first step in the demodulation process is down conversion from a real passband waveform to a complex I/Q baseband waveform. This process involves mixing the real-valued passband waveform with a locally generated carrier tone, followed by lowpass filtering to generate the I/Q baseband waveform.
DSP	**Digital Signal Processing**—The computation of signal or system transfer characteristics using numeric algorithms. Examples of areas where DSP techniques may be applied include: digital filters, echo detection or echo cancellation, speech synthesis, FFT for spectrum analysis, correlation computations, imagine recognition, and servo-feedback control.

E

encoding	A data source generates the information signal sent to a particular receiver. This signal may be either an analog signal, such as speech, or a digital signal, such as a binary data sequence. The information signal is typically a baseband signal represented by a voltage level.
equalization	In an adaptive feed-forward equalizer, the taps of a filter that acts as an equalizer that continuously adapts its coefficients to compensate for the action of the channel filter. At the start of the equalization process, you typically supply training bits to train the equalizer. After training, the equalizer switches to decision-directed feedback mode, where the

equalizer trains itself based on its own decisions.

F

fetch	Process that transfers data from device onboard memory to PC memory.
filtering	In a digital communication system, digital information can be sent on a carrier through changes in its fundamental characteristics such as phase, frequency, and amplitude. In a physical channel, these transitions can be smoothed, depending on the filters implemented during transmission. In fact, filters play an important part in a communications channel because
FIR filter	**Finite Impulse Response (FIR)** is a term that describes a filter with no feedback elements, hence, its impulse response is finite. In contrast, IIR (infinite impulse response) circuitry does use feedback. FIR filters can be implemented by using analog or digital shift registers, or by using software algorithms. For a tutorial of IIR and FIR filters, search for *digital filter* in the NI Developer Zone.
FFT for	The **Fast Fourier Transform (FFT)** is an efficient algorithm often used spectrum analysis.
FM	**Frequency Modulation (FM)** is a process that varies the frequency of a sinusoidal carrier wave from a center frequency by an amount proportional to the instantaneous value of the message signal. In FM, the center frequency is the carrier frequency. 
FPGA	A **Field-Programmable Gate Array (FPGA)** is a semi-conductor device which contains a large quantity of gates (logic devices), which are not interconnected, and whose function is determined by a wiring list, which is downloaded to the FPGA. The wiring list determines how the gates are interconnected, and this interconnection is performed dynamically by turning semiconductor switches on or off to enable the different connections. Search for ***FPGA*** in the NI Developer Zone for a more complete explanation.
frequency	**Frequency** refers to a basic unit of rate measured in events or oscillations

	per second. Frequency also refers to a number representing a specific point in the electromagnetic spectrum. Frequency is measured as the number of cycles per unit time. The SI unit of measure for frequency is Hertz (Hz) where 1 Hz equals one cycle per second.
frequency span	Typically refers to a range of frequencies, for example from 10 kHz to 20 kHz. This term often describes the range that an instrument, such as a spectrum analyzer, is set to measure, or the range over which a set of frequencies of interest are located. For example, an FM modulated signal may cover a span from 100.5 MHz to 100.7 MHz.

G

gain	The factor by which a signal is amplified, often expressed in dB. **Gain** as a function of frequency is commonly referred to as the magnitude of the frequency response function.

H

Hertz, Hz	**Hz** is the SI unit for measurement of frequency. One **Hertz** (Hz) equals one cycle per second. It can be used to represent the number of scans read or updates written per second.

I

IIR filter	**Infinite Impulse Response Filter**—A filter that is designed using feedback or storage elements so that its impulse response will, in principle, last forever. In contrast, FIR filters have a finite impulse response.
impairments	All transmission media (including wireless, fiber optic, and copper) introduce some form of distortion (**impairments**) to the original signal. Different types of channel models have been developed to mathematically represent such real-world distortions.
in-phase signal	The **I (in phase)** component of a signal is typically the real component of a complex-valued signal.
I/Q data	The translation of the magnitude and phase data of a signal from a polar coordinate system to a complex Cartesian (X,Y) coordinate system. The I (In-phase) component is typically the real-valued component of the signal, while the Q component is typically the imaginary component of the signal.
I/Q modulation	**In-Phase/Quadrature Modulation (I/Q modulation)** is a modulation technique where a signal is modulated by two signals 90 degrees out of phase with each other.
I/Q signal	A control signal for changing an RF carrier signal.
IF the	**Intermediate Frequency (IF)**—In radio receivers or spectrum analyzers, original high-frequency signal is often mixed to an intermediate frequency before demodulation.

information signal	Contains the data for transmission. The information signal is used to modulate the carrier wave to create the modulated wave for transmission. The information signal data is recovered from the modulated wave by a process of demodulation. The information signal is often referred to as the baseband signal or *message signal*.
impedance	The electrical characteristic of a circuit that opposes flow of current through that circuit. It can be expressed in ohms and/or capacitance/inductance.
intersymbol interference	In telecommunication, **intersymbol interference** (**ISI**) is a form of distortion of a signal in which one symbol interferes with subsequent symbols. This is an unwanted phenomenon as the previous symbols have similar effect as noise, thus making the communication less reliable. ISI is usually caused by multipath propagation or the inherent non-linear frequency response of a channel causing successive symbols to "blur" together.[4]
ISM	The **Industrial, Scientific and Medical (ISM)** bands are low-power radio frequencies for industrial, scientific, and medical processes, and license-free wireless communication.

L

Local Oscillator (LO)	**Local Oscillator** refers to an internal oscillator in a radio, receiver, or instrument that tunes the frequency of the device. The local oscillator is mixed with a fixed frequency oscillator, which generates a sum and difference component.

M

message signal	Contains the data for transmission. The **message signal** is used to modulate the carrier wave to create the modulated wave for transmission. The message signal data is recovered from the modulated wave by a process of demodulation. The message signal is often referred to as the baseband signal or *information signal*.

MIMO	**Multiple input, multiple output**—A measurement technique, most often used in acoustics and vibration, to identify signal paths and frequency response functions from multiple inputs to multiple outputs. Associated with this measurement class are the techniques of partial and multiple coherence, which help identify which parts of an output signal are due to a specific input signal or combinations of signals.
modulation	A process, or the result of a process, by which characteristics of a carrier wave are altered according to information in the baseband signal to generate a modulated wave that is transmitted. In a diagram of the components of a Software Defined Radio system, the modulation block converts the information signal bit stream into in-phase (I) and quadrature phase (Q) data components. This block typically also involves pulse shaping to minimize intersymbol interference and reduce bandwidth.
P	
passband	The range of frequencies that a device can properly propagate or measure.
phase	The fraction of the wave cycle which has elapsed relative to the origin[5]
phase-locked loop (PLL)	An electronic circuit that controls an oscillator so that the circuit maintains a constant phase angle relative to a reference signal.
PPS	Pulse Per Second
pulse-shaping filter	By applying a pulse-shaping filter to the modulated sinusoid, sharp transitions in the signal are smoothed and the resulting signal is limited to a specific frequency band. In addition, pulse-shaping filters are applied to communication signals to reduce intersymbol interference.

Q

quadrature signal	The Q (**quadrature**) component of a complex-valued signal is typically the imaginary-valued component of the signal.
QAM	**Quadrature-Amplitude Modulation (QAM)** is a form of quadrature modulation in which the two carriers are both amplitude-modulated.

R

radio frequency (RF)	Refers to the radio frequency range of the electromagnetic spectrum. RF is often used to describe a range of sub-infrared frequencies from the tens of MHz to several GHz.
reference clock	Clock to which a device phase locks another, usually faster, clock. A common source for the Reference clock is the 10 MHz oscillator present on the PXI backplane.
RX, Rx	Receive data or signals. RX refers to the hardware receiver; Rx refers to receive operations in software.

S

sample rate	The rate at which a device acquires an analog signal, expressed in samples per second (S/s). The sample rate is typically the clock speed of the analog-to-digital converter (ADC).
SDR	**Software-Defined Radio**
SerDes	**Serializer/Deserializer**
SFDR	**Spurious Free Dynamic Range**—The separation or distance, expressed in dB, from the amplitude of the fundamental frequency and the next highest spur.
SMA	A small type of threaded coaxial signal connector typically used in higher frequency applications.
symbol rate	Expresses the number of symbols transmitted per second (symbols/s). To convert symbol rate into bit rate, which expresses the number of bits transferred per second, multiply the symbol rate by the number of bits per symbol used in the digital modulation scheme of interest. Symbol rate is

also known as *baud rate*.

T

TTL	**Transistor-Transistor Logic**—A digital circuit composed of bipolar transistors wired in a certain manner. A typical medium-speed digital technology. Nominal TTL logic levels are 0 and 5 V.
TX, Tx	Transmit data or signals. **TX** refers to the hardware transmitter; **Tx** refers to transmit operations in software.

U

UHD	**Universal Hardware Driver**, a software driver that is called on by the LabVIEW driver to provide control over a connected USRP device.
USRP	**Universal Software Radio Peripheral**—A computer-hosted hardware peripheral used to create software-defined radio systems.
upconverter	A signal conditioning device that converts a specific band of IF frequencies to high-frequency (RF) signals.
Upconversion	In a diagram that shows the components of a Software Defined Radio, the baseband modulated signal undergoes analog up conversion to frequency-translate the signal to the RF frequency at which the signal is transmitted.

V

VDC	**Voltage Direct Current**

REFERENCES

- **Universal Software Radio Peripheral on Wiki,**

http://omsec.com/wiki?UniversalSoftwareRadioPeripheral

- **Rf Sections 4 USRP on Wiki,**

http://comsec.com/wiki?RfSections4USRP

- **Matt Ettus USRP User’s and Developer’s Guide,**

http://home.ettus.com/usrp/usrp guide.html

- **USRP Install on Wiki,**

http://comsec.com/wiki?UsrpInstall