****

**Instrument Control Project**

**(TCP/IP based control and streaming)**

**Proof of concept document**

Contents

[1. Introduction 3](#_Toc454569576)

[2. Demonstration objectives 4](#_Toc454569577)

[2.1. Requirements, list of features: 4](#_Toc454569578)

[3. Required Hardware: 5](#_Toc454569579)

[4. Required Software 6](#_Toc454569580)

[4.2. Delivered Software package 8](#_Toc454569581)

[5. Remote Control – Description of commands 8](#_Toc454569582)

[5.1. Introduction 8](#_Toc454569583)

[5.2. Notation 8](#_Toc454569584)

[5.3. Common Commands 8](#_Toc454569585)

[~~\*IDN?~~ 8](#_Toc454569586)

[~~5.4.~~ ~~ABORt subsystem~~ 8](#_Toc454569587)

[5.5. SENSe subsystem 8](#_Toc454569588)

[SENSe:FREQuency Subsystem 8](#_Toc454569589)

[SENSe:BANDwidth subsystem 9](#_Toc454569590)

[SENSe:RLEVel subsystem 9](#_Toc454569591)

[5.6. DISPlay subsystem 9](#_Toc454569592)

[VBW – Video Bandwidth 9](#_Toc454569593)

[DISP:AVERage:MODE 10](#_Toc454569594)

[DISP:AVERage:SAMPles 10](#_Toc454569595)

[DISP:TRACe 10](#_Toc454569596)

[5.7. MEASurement subsystem 10](#_Toc454569597)

[MEAS:MEASure 10](#_Toc454569598)

[MEASurement:CONTinuous {0,1} 10](#_Toc454569599)

[MEASurement:ABORt 10](#_Toc454569600)

[5.8. Other subsystems 10](#_Toc454569601)

[6. Remote Control – Examples 11](#_Toc454569602)

[6.1. Python example (test\_tcp.py) 11](#_Toc454569603)

[6.2. SCPI scripting example (pxi-01.txt, pxi-02.txt, pxi-03.txt): 11](#_Toc454569604)

[7. Streaming channel 12](#_Toc454569605)

[7.1. Data format 12](#_Toc454569606)

[7.3. Example 13](#_Toc454569607)

[Sending 10k samples, all zeros 13](#_Toc454569608)

[F0=1, df=1, then 11000 pcs of “-2”: 14](#_Toc454569609)

[8. Demo 15](#_Toc454569610)

[9. ToDo list 15](#_Toc454569611)

|  |  |  |
| --- | --- | --- |
| Version | Date | Author |
| Ver1 | May 13 2016 | Peter Vago |
| Ver3 (delivered) | June 25 2016 | Peter Vago |
| Ver4 | June 27 2016 | Peter Vago |
| Ver5 (netcat client example added | Aug 22 2016 | Peter Vago |
|  |  |  |

# Introduction

This document was created to give a quick overview of designed features and implementation, timeline of this Proof of Concept (PoC) project.

PoC implementation and deliverables can be distributed under the NI sample code license. (<http://ni.com/samplecodelicense> )

PoC deliverables are:

* Documentation
* Reference design\*
* Demonstration (remotely by Systems Engineer)

\*Reference design/ ref. application:

Open source code shared with the NI platform community (e.g. published on ni.com, LV Tools Network or a public VIPM repository), intended to

    -Provide examples of how to use specific features, or develop parts or the whole of an application

    -Provide reference code that a developer may use as a template or higher-level starting point for developing all or part of a new application. Reference code is intended to be copied and modified for specific applications.

    -Promote and teach best practices in software design and architecture

This code is distributed under the Sample Code License.

A reference application may be distributed as a Sample Project so that developers can easily create a new instance of the application.

# Demonstration objectives

The Demonstration Application intends to demonstrate how NI PXI Instruments can be controlled by clients over TCP/IP and what the streaming capabilities of NI PXI systems are over network.

PoC will deliver a single connection, single device control channel.

Device will be an NI VSA (Vector Signal analyzer), e.g. NI 5663 or 5665.

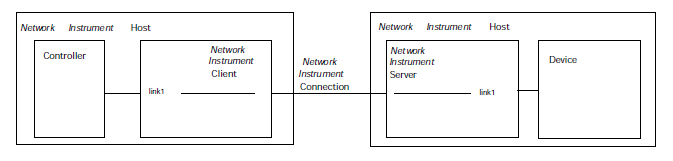
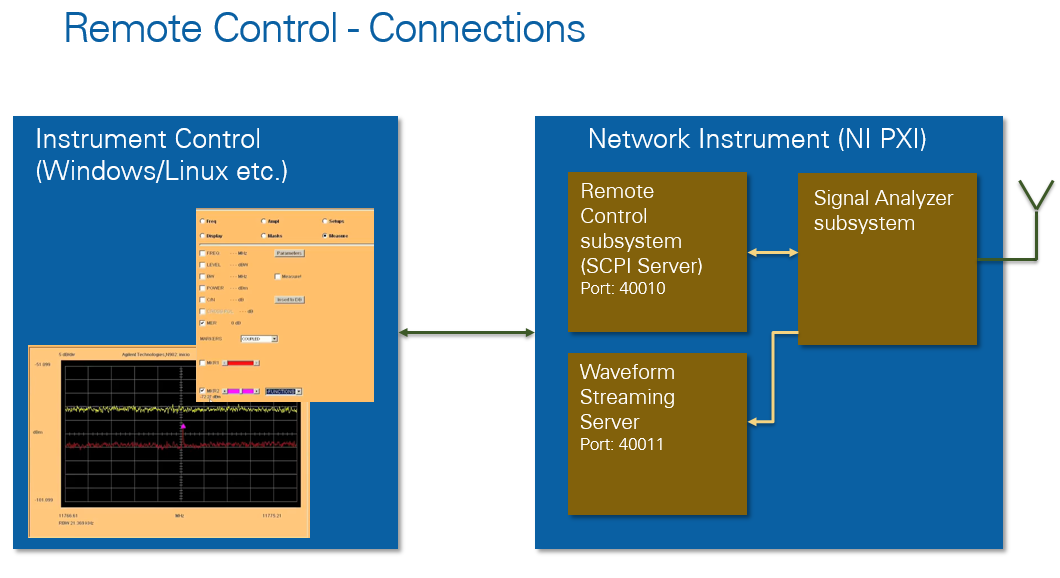


Figure 1: Single link, single device (picture from VXI-11 std)

Protocol will be designed according to VXI-11 standard, probably with some similarities to SCPI-99 protocol.



1. ábra: Connections, ports

## Requirements, list of features:

Instrument server control capabilities:

* Instrument (VSA)
  + Open session
  + Close session
  + Set/Get Frequency (center freq.)
  + Set/Get Span or Set/Get Start and Stop freq.
  + Set/Get RBW (Resolution bandwidth)
  + Run/Stop Acquisition
  + Run Measurements
    - Power level of highest peak in the Span
    - Peak frequency
  + Get Error and Status messages
* Streaming
  + Set server port (PXI will be the server/listener)
  + ~~Set IP address (it will be fixed for first..)~~
  + Initiate session
  + Release session

# Required Hardware:

- PXIe chassis + Controller

- 1xNI VSA\*\* (NI 566x) -> Important: Name it as “NI\_VSA” in MAX. This is hardcoded in the server software. (later improvement ->this can be read from INI file.)

- (Optional: NI 5690 or 5691 Preamplifier)

- RF signal source or antenna

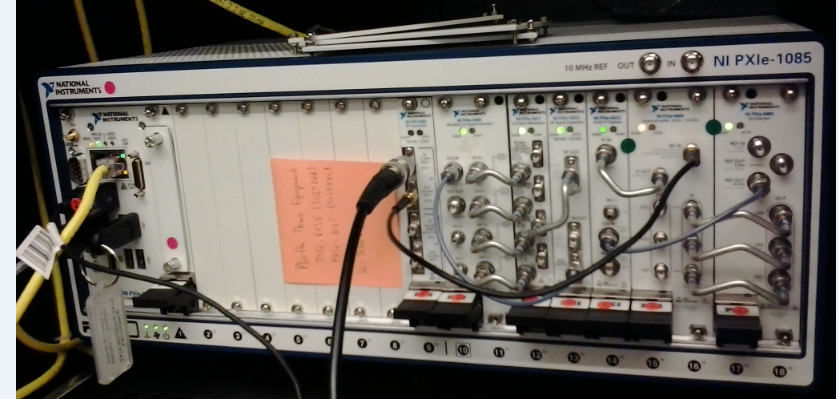
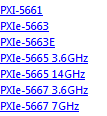
****

Figure 2: Demo System (PXIe with VSA and VSG)

\*\*Supported VSA types (all NI-RFSA based devices):





# Required Software

* Server:
  + LabVIEW 2014 Runtime-Engine (for executable)
  + LabVIEW 2014 (for source code)
    - A few other LabVIEW libraries are needed (OpenG) and others. I’ll try packing everything together in a single project…

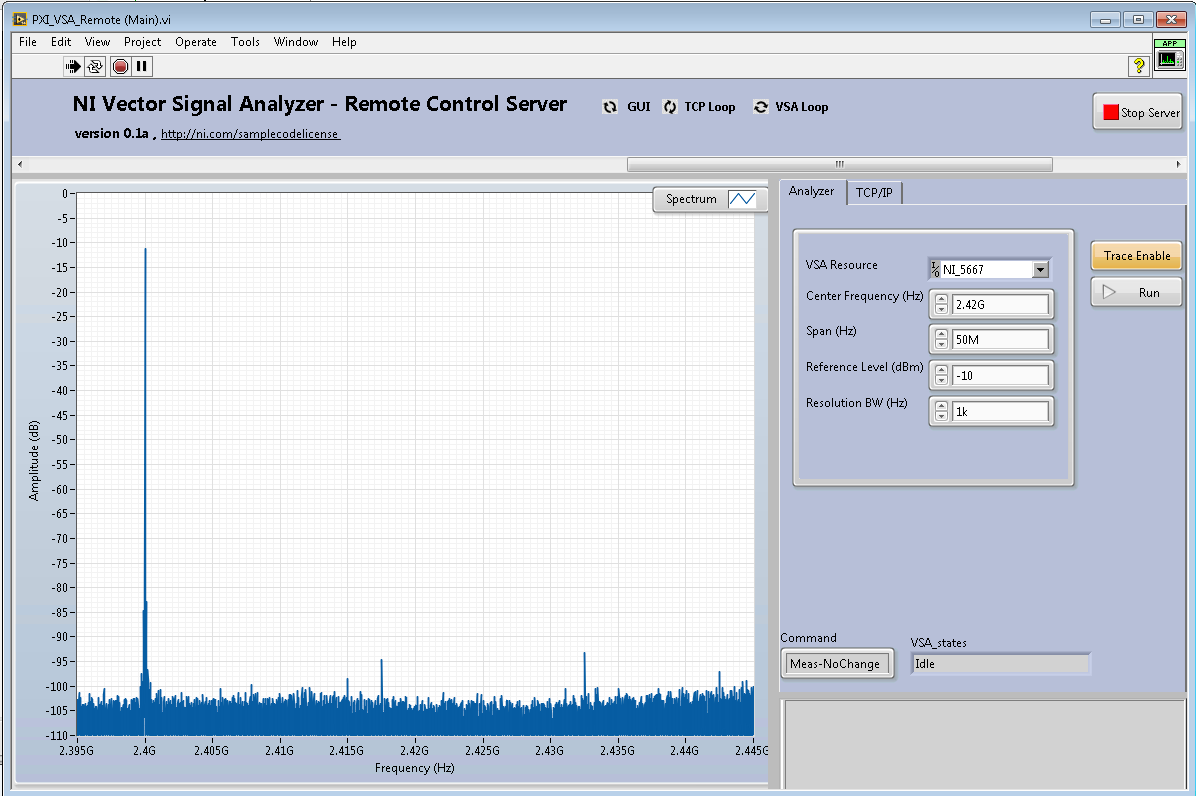


Figure 3: Front Panel of server application

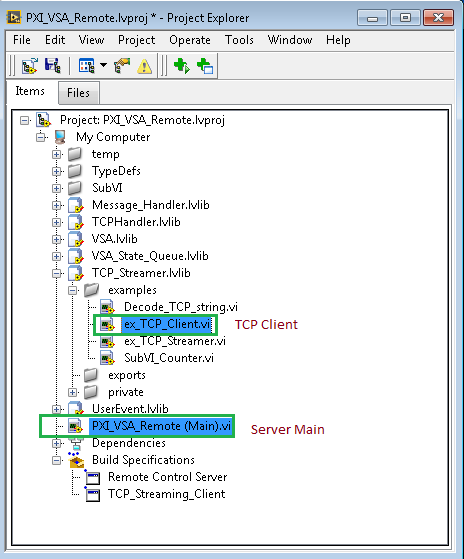


Figure 4: LabVIEW project of Server application

* Client:
  + LabVIEW 2014 for LabVIEW TCP Streamer client example (ex\_TCP\_Client.vi), or LV RunTime-Engine for executable
  + Python 3.x for Python client example (+pyvisa library, +NI-VISA driver OR pyvisa-py backend installed on client machine)

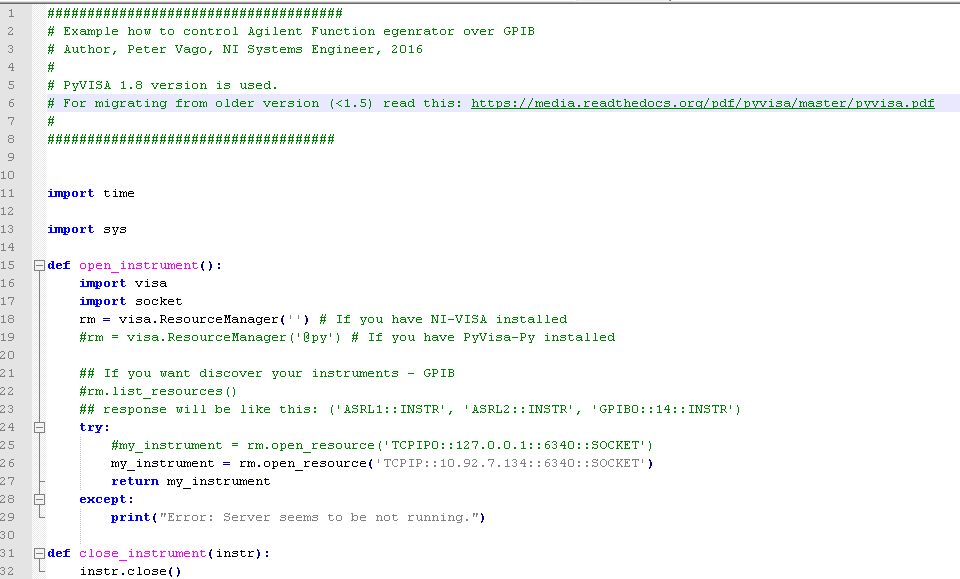
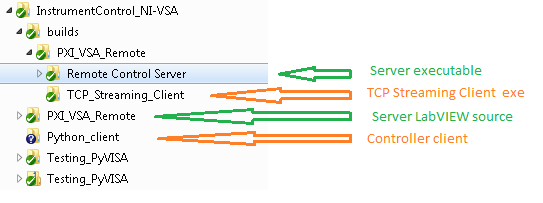


Figure 5: Python client example (snippet)

## Delivered Software package



# Remote Control – Description of commands

Control port: 40010

## Introduction

## Notation

## Common Commands

### ~~\*IDN?~~

Not implemented yet.

## ~~ABORt subsystem~~

Not implemented yet.

## SENSe subsystem

### SENSe:FREQuency Subsystem

**SENSe:FREQuency:CENTer 0 to fmax**

This command defines the center frequency of the VSA or the measuring frequency for span = 0.

Example:

"FREQ:CENT 100MHz"

Characteristics:

\*RST value: fmax/2 with fmax= maximum frequency

SCPI: conform

**SENSe:FREQuency:SPAN 0 to fmax**

In analyzer mode, this command defines the frequency span.

Example:

"FREQ:SPAN 10MHz"

Characteristics:

\*RST value: fmax with fmax = maximum frequency

SCPI: conform

### SENSe:BANDwidth subsystem

**SENSe:BANDwidth|BWIDth[:RESolution] 10 Hz to max**.

This command defines the resolution bandwidth.

Analog resolution filters of 10 Hz to 20 MHz in 1, 2, 3, 5, 10 steps are available. Additionally there is a

50 MHz resolution bandwidth. These filters are implemented as 5-circuit LC filters in the range from

300 kHz to 10 MHz and as digital filters with analog characteristic in the range of 10 Hz to 100 kHz.

**Example:** "BAND 1MHz"

’Sets the resolution bandwidth to 1 MHz

**Characteristics:** \*RST value: - (AUTO is set to ON)

SCPI: conform

### SENSe:RLEVel subsystem

**SENSe:RLEVel -100 to 10**

This command defines the Reference Level in dBm

## DISPlay subsystem

### VBW – Video Bandwidth

VBW is emulated in Vector Signal Analyzers: <http://www.ni.com/tutorial/13869/en/>

This example is used for implementing it: <https://decibel.ni.com/content/docs/DOC-22813>

(VBW Example.vi

When trying to measure signals in the presence of noise, the [Video Bandwidth (VBW)](http://digital.ni.com/express.nsf/bycode/ex867f" \t "_blank) filter can be used to reduce the variation in the noise while leaving the signal of interest unaffected. In order to emulate a VBW filter on hardware that does not have a physical filter, VBW can be applied in software using averaging dictated by the following equation:

N = [(k\*RBW/VBW)^p + 1]^(1/p)

Where

k: .536

p: 1.275

N: Number of Averages

)

**DISPlay:VBW 0 to max**

Values: 0 ->bypass, >0 -> values in Hz

### DISP:AVERage:MODE

**DISPlay:AVERage:MODE {OFF,RMS,PEAK}**

Off: Disabled

RMS: RMS

PEAK: Peak Hold

### DISP:AVERage:SAMPles

Number of averages. Minimum 1, maximum 100 (not limited on server, but not tested with >100)

**DISPlay:AVERage:SAMPles 1 to max**

DISP:TRACe 1

Used for Enable trace on Soft Front Panel. Disabling trace can lead to higher performance.

Enabled: 1 , Disable: 0 (Default: 0)

**DISPlay:TRACe {0,1}**

## MEASurement subsystem

### MEAS:MEASure

Taking measurement (and streaming data through Streaming channel)

Measurement can be single-shot or continuous. For switching this option, see MEAS:CONT command.

**MEAS:MEASure**

**(MEAS:STARt)**

### MEASurement:CONTinuous {0,1}

Used for enabling Continuous measurement.

If this option is disabled, then instrument will take only one measurement after sending MEAS:MEAS command.

Enable: 1, Disable: 0 (Default 0)

**MEAS:CONTinuous {0,1}**

### MEASurement:ABORt

This is used only in continuous mode. After sending MEAS:MEAS the instrument will start measuring continuously. Measurement can be stopped by sending this MEAS:ABORt command.

**MEAS:ABORt**

**(MEAS:STARt)**

## Other subsystems

# Remote Control – Examples

## (Python install)

### Windows:

* Install python: from Npack package manager of from the official webpage
  + Add environment variables, as written here: <http://stackoverflow.com/questions/3701646/how-to-add-to-the-pythonpath-in-windows-7>
    - Create new variable Name: “PY\_HOME”, Variable Value: “C:\path\to\python\version”
    - Add the following to the existing „Path” variable: „%PY\_HOME%;%PY\_HOME%\Lib;%PY\_HOME%\DLLs;%PY\_HOME%\Lib\lib-tk; „
    - Then you can enter “python” to command line and it will start python shell (type exit() to exit)
* Install Python libraries with pip (Pyhton Installer Package)
  + Run cmd as administrator
  + Install pip : “python get-pip.py” (get-pip.py is attached to delivered zip package)
  + Install VISA library. You have 2 options:
    - 1) Install py-visa: “python -m pip install pyvisa” (this will download packages from the Internet) . This is based on NI VISA driver (supported for Windows and for some linux distros)
    - 2) Install pyvisa-py: “python -m pip install pyvisa” (this will download packages from the Internet)

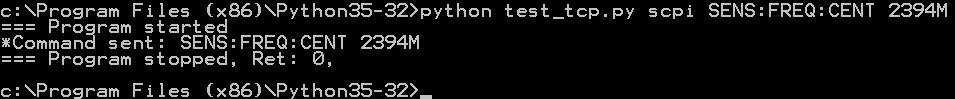
## Python example (test\_tcp.py)

Server has to be running first.



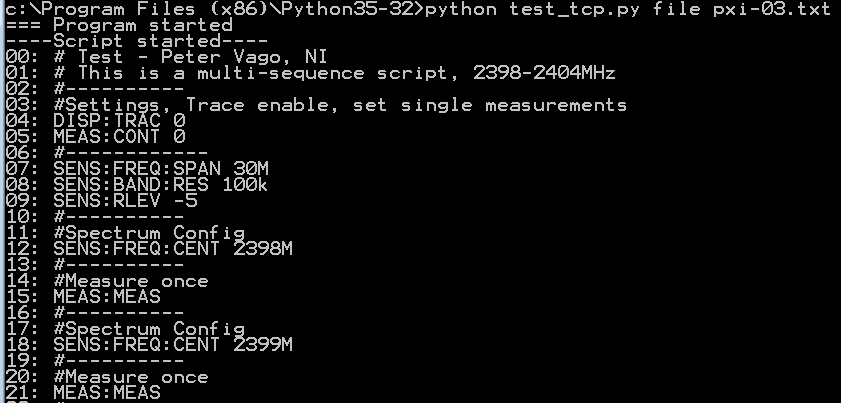
Syntax for sending SCPI commands:

python test\_tcp.py scpi <string [parameter]>



Syntax for running scripts:

python test\_tcp.py file <file\_path>



## SCPI scripting example (pxi-01.txt, pxi-02.txt, pxi-03.txt):



# Test - Peter Vago, NI

# This is a multi-sequence script, 2398-2404MHz

#----------

#Settings, Trace enable, set single measurements

DISP:TRAC 1

MEAS:CONT 0

#------------

SENS:FREQ:SPAN 30M

SENS:BAND:RES 100k

SENS:RLEV -5

#----------

#Spectrum Config

SENS:FREQ:CENT 2398M

#----------

#Measure once

MEAS:MEAS

#----------

#Spectrum Config

SENS:FREQ:CENT 2399M

#----------

#Measure once

MEAS:MEAS

#----------

#Spectrum Config

SENS:FREQ:CENT 2400M

#----------

## 6.3 Putty example

I do not recommend using Putty for testing. Reason: http://the.earth.li/~sgtatham/putty/0.61/htmldoc/Chapter3.html#using-rawprot

## 6.4. netcat

netcat is a default tool on most Linux distributions. Can be installedon Windows too : http://kudithipudi.org/2011/07/14/how-to-use-netcat-nc-on-windows-7/

Example:

Single command:

echo "SENS:FREQ:CENT 2450M" | nc <server ip> <server port>

Multiple commands:

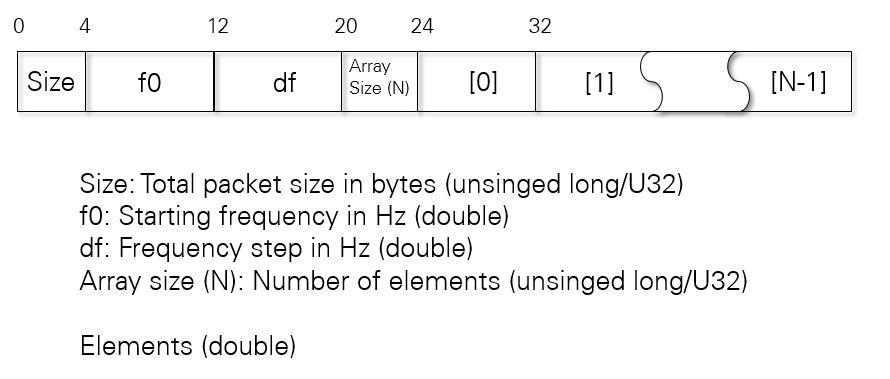
1. write your commands into a text file (my\_commands.txt)

2. cat my\_commands.txt | nc <server ip> <server port>

# Streaming channel

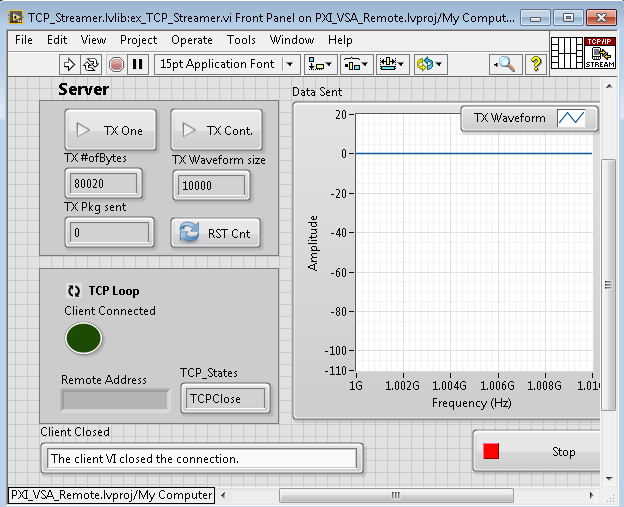
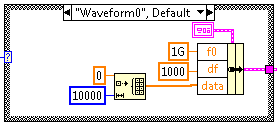
Port:40011

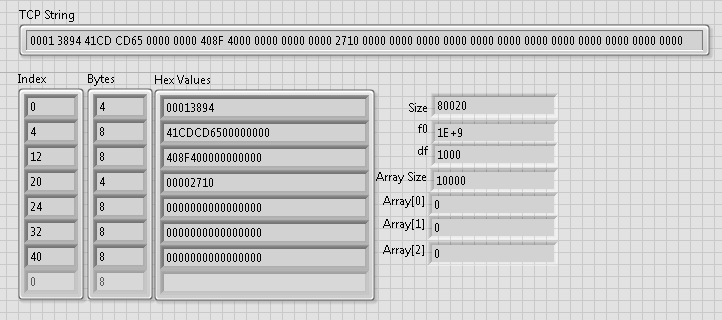
## Data format



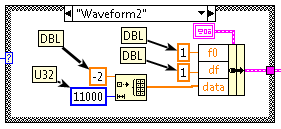
## Example

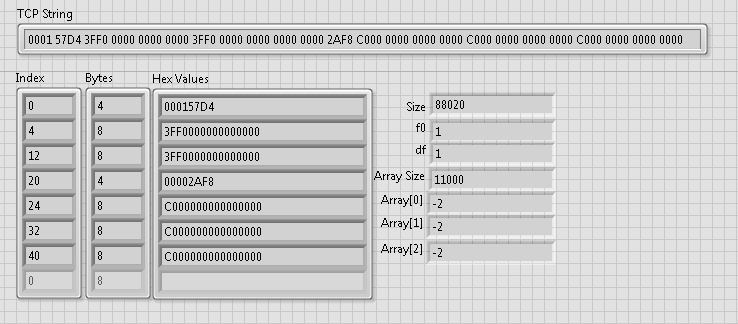
### Sending 10k samples, all zeros





### F0=1, df=1, then 11000 pcs of “-2”:





References:

* <https://zone.ni.com/reference/en-XX/help/371361J-01/lvconcepts/how_labview_stores_data_in_memory/>
* <https://en.wikipedia.org/wiki/Double-precision_floating-point_format>

# Demo

First video demo is uploaded to Youtube.com : <https://youtu.be/vN_r_bdAizM>

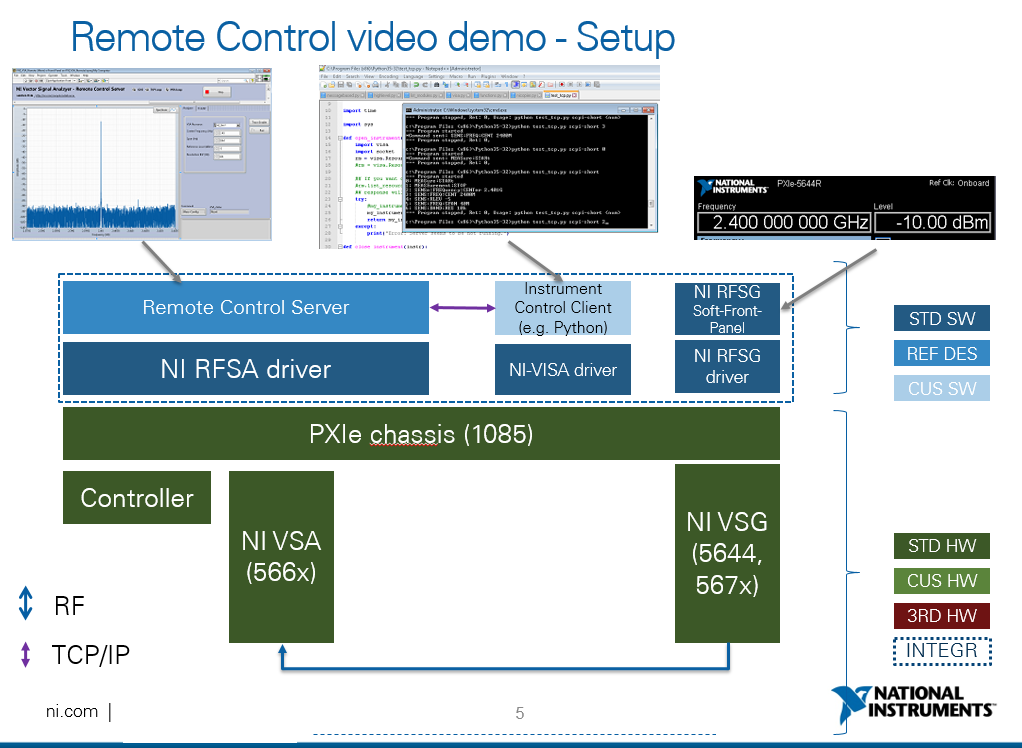
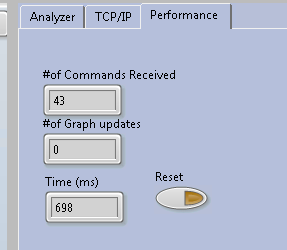


Figure 6: Setup that used for video demo

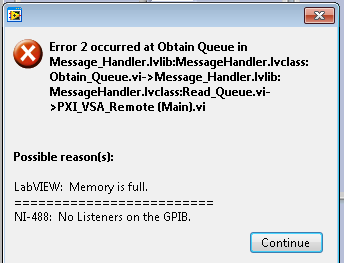
# ToDo list

* Read initial values form INI file
* Design: Better queue organization (classes)
* Finishing: TCP streaming of Spectrum data
* Testing/finishing queries (“\*IDN?”, all other parameters that is defined in this document)
* Compiling and testing server executable
* Performance measurement (pxi-03.txt script -> 8 steps)

698ms (Trace disabled) and 810ms (Trace enabled)



* Memory leak detected: Queue references were not closed properly. Fixed (June 27)



Document created in June 2016.

Peter Vago – Systems Engineer (peter.vago@ni.com, + 36 30 493 4016)  
National Instruments Europe