This user guide describes how to use the doppler emulator example program and provides an overview of the software architecture.
Doppler Effect Overview

“Doppler’s effect can be defined as an observed change in frequency of a wave when an observer and source have a relative motion between them. In other words, it is an increase or decrease in the frequency of waves as the source and observer move toward or away from each other.

The observed changes in frequency associated with Doppler Effect can be explained as follows:

When the source is moving toward the observer, each consecutive wave is emitted from a position closer to the observer than the previous wave. So when travelling, the waves are seemingly grouped together, and the time between the arrival of successive wave crests to the observer is reduced (thus reducing the wave length), causing an increase in the frequency, as the velocity is constant.

However, when the source is moving away from the observer, each wave is emitted from a position further away from the observer than the previous wave, making the waves more spread, thereby increasing the wavelength and reducing the frequency.

When the source and observer are moving, the following equation can be used:

\[ f = \left( \frac{c + V_s}{c + V_0} \right) f_0 \]

Where, \( f \) is the observed frequency, \( f_0 \) is the actual frequency, \( C \) is the velocity of waves in the medium, \( V_s \) is the velocity of the source and \( V_0 \) is the velocity of the observer” *

Doppler effect is important for many applications such as detecting the speed of motion of stars and galaxies in astronomy field, medical imaging to detect vascular anomalies by measuring the speed of blood flow and in radar applications where a signal is sent towards a moving target to detect its speed. Doppler effect is also widely used in channel sounding applications to study the characteristics of a channel and mitigate its effect on the signal due to many physical phenomena such as reflection, refraction, diffraction and shadowing.

In this example program, we are emulating doppler effect on a received signal on 4 different paths where each path shifts the frequency \( f_0 \) by a value configured by the user \( \Delta f \), and then the signals with doppler shift are added to the original direct path then transmitted back to an external medium.

The doppler emulator can be used to test the behavior of receivers used for instance in radar and imaging applications with respect to a moving source, and to emulate effects of channel or emulate a moving target.

This example code is designed for the NI Vector Signal Transceiver (VST) 5646 where the doppler emulator is written on FPGA.

An RF signal is received by the Rx port of the 5646 and it passes through the DSP chain on FPGA where the doppler IP is applied to the signal. Then the signal is transferred to the Tx path to be played out and received by an external analyzer.

*Extracted from Doppler Effect Wikilectures [https://www.wikilectures.eu/w/Doppler_Effect](https://www.wikilectures.eu/w/Doppler_Effect)
Doppler Emulator User Interface

The Host VI “Doppler Emulator (Host).vi” allows you to configure the device name and RF parameters such as carrier frequency, IQ rate and Resolution Bandwidth as well as the Doppler parameters for each path.

- This example code supports up to 4 paths only. For each path you can configure the frequency shift from the main frequency and the amplitude attenuation of that path in dB also with respect to the received signal power.
- A path can be enabled or disabled by clicking on the enable button, and the doppler parameters can be modified on the fly.
- A graph will show what will the doppler signal look like according to the configured parameters.
- The maximum frequency supported by the doppler shift depends on the span that you configure.
  \[ \text{Max doppler} = \frac{\text{span}}{2} \]
- The IQ rate (span*1.25) is the same for both receiver and transmitter paths.
- If you run the host and you see the full span on the graph, you might need to zoom in on the graph to see the different taps enabled or zoom out if you configure a frequency shift outside your display span.
Pay attention to the resolution bandwidth you use. If your RBW is close or higher than the frequency offset that you configure you will see the main signal and the doppler path as a same frequency. Set your RBW lower than your frequency offset to separate the 2 signals.

You will need to generate a signal externally and wire it to the Rx port of the VST 5646, then capture the VST generated signal via the Tx port by an external analyzer. You can use another VST or any generator/analysing. Make sure to set the parameters properly on your analyzer (span, RBW) to capture the different paths.
Doppler Emulator Host Block Diagram

The Block Diagram consists of 2 main paths, the Tx and Rx.

Both paths connect to the same FPGA image and send the user configuration for both Tx and Rx.

The configured doppler parameters are transformed to fixed point on the host before being sent to the FPGA. The received signal on the FPGA is shared with the host after applying Doppler IP via a DMA FIFO and then the doppler signal is displayed on the HMI graph.

Doppler Emulator FPGA Block Diagram

The FPGA code is based on the VST Instrument Design Library Template which contains proper configuration for the bus registry and the DSP chains for both Tx and Rx.

Below you will find snapshots of the main loop with both Tx and Rx DSP chains. First, signal is received and downconverted, then doppler IP is applied before being written to both a DMA to share the signal with the host and a target scoped FIFO to share the signal with the Tx path.

The Tx path receives the signal via the target scoped FIFO before being upconverted and sent via the front end.

A proper synchronization is implemented between the Tx and Rx to avoid intermodulation distortion caused by the copy of the signal being added to the main signal, as well as timing violations.
The doppler IP consists of 4 frequency shifters and digital gain added on a copy of the original signal each. Then the direct path is added to the 4 doppler paths.
You can consider removing the direct path if you wish to apply only the doppler shift. You can also add more paths as you wish on the FPGA. The limitation will be the utilization of FPGA resources and meeting timing constraints during the FPGA compilation.

The current resources utilization for the Doppler Emulator FPGA code are the following:

```
5646:
----------------------------------------
Device Utilization
----------------------------------------
Total Slices: 60.6% (22832 out of 37680)
Slice Registers: 26.9% (81045 out of 301440)
Slice LUTs: 41.4% (62415 out of 150720)
Block RAMs: 39.8% (449 out of 1188)
DSP48s: 83.3% (640 out of 768)
----------------------------------------
Timing
----------------------------------------
ReliableClock (used by non-diagram components): 125.00 MHz (141.00 MHz maximum)
40 MHz-Onboard Clock: 40.00 MHz (45.00 MHz maximum)
Data Clock 1: 250.00 MHz (251.95 MHz maximum)
Data Clock 2: 125.00 MHz (127.83 MHz maximum)
Data Clock 3: 250.00 MHz (251.95 MHz maximum)
100 MHz: 100.00 MHz (130.80 MHz maximum)
133 MHz: 133.00 MHz (126.30 MHz maximum)
----------------------------------------
```

Compilation success rates have been about 90% for the NI PXIe-5646R, NI PXIe-5645R, and the NI PXIe-5646R. As long as the code is properly pipelined, timing should not be an issue.

**Figure 8: FPGA Resource utilization**

**Required Hardware and Software**

**Required Hardware:**

This example code is built for the NI PXIe-5646 but you can try to import it on any other VST by making the necessary changes to adapt to the module data clock and number of samples per cycle.

You will also need another VST or VSA+VSG to feed a signal into the VST5646 and receive back the doppler shifted signal.

**Hardware setup recommendation:**

- 2x NI VST 5646, or 1xVST 5646 + 1xVST5840 (you need a digitizer with instantaneous BW equal or larger than that of VST5646)
- PXIe-1092 chassis
- PXIe-8880 controller with Win 10

**Figure 9: Suggested System Configuration**
Required Software
This code uses the NI VST Instrument Design Libraries which are supported by LabVIEW 2015 the latest. You will need below SW:

- LabVIEW 2015
- LabVIEW FPGA Module 2015
- NI-RFSA 2015 or later
- NI RFSG 2015 or later
- NI VST Instrument Design Libraries 2015

Example Code Specifications
- Fading Model: Pure Doppler
- Max Doppler Shift: 100MHz
- Number of Paths: 4 + Direct Path
- Max Bandwidth: 200MHz
- Dynamic Range: 80dB
- Single Input Single Output Configuration

Feedback
For your feedback on this example code (bug reports and further improvements, please post a comment on the community.