# Summary - Lime TX Proxy

| | |
|---|---|
| Name | lime_tx_proxy |
| Worker Type | Proxy |
| Version | v1.3 |
| Release Date | February 2018 |
| Component Library | ocpi.assets.devices |
| Workers | lime_tx_proxy.rcc |
| Tested Platforms | xilinx13_3, CentOS 7 (via alst4/Zipper), CentOS 6/7 (via ml605/Zipper for HPC and LPC FMC slots) |
| Slave Worker | lime_tx.hdl |

# Functionality

This control proxy is designed to allow the user of the proxy to set more user friendly properties than the register map on the LMS6002D Transceiver. Only the control of the TX portion of the LMS6002D Transceiver is encompassed in this worker.

# Worker Implementation Details

## lime_tx_proxy.rcc

A diagram of the transmitter in the Lime Microsystems LMS6002D can be seen in figure 2. The FPGA provides complex samples to the LMS6002D on a 12 bit multiplexed bus, and analog IQ signals are generated by on chip DACs. These signals then pass through a low pass filters and a programmable gain amplifier. After the first amplifier stage, DC offset is inserted in the IQ path in order to cancel the LO leakage. The IQ signals are then mixed with a PLL output to produce a modulated RF signal. This RF signal is then split and amplified by two separate controllable gain amplifiers, only one of which can be active at any given time, limiting the device to a single transmit channel. After the second amplifier stage, the RF signal is routed to an output pin on the transceiver.

The features described above are controllable via a SPI interface on the LMS6002D. This proxy is responsible for translating its properties (as described in the Lime datasheet) into the required SPI reads and writes and controlling the worker which performs the SPI transactions.

# Block Diagrams

## Top level

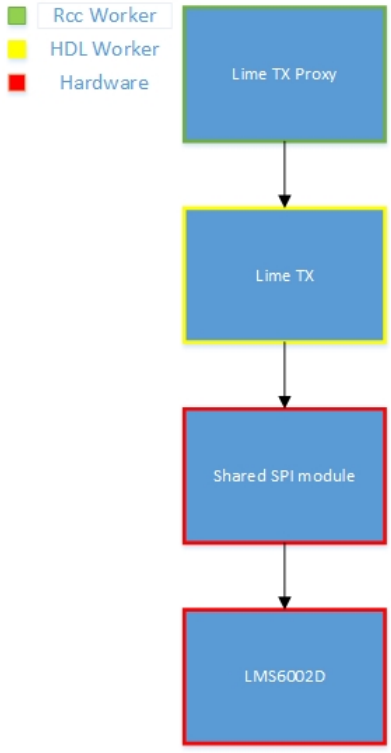


Figure 1: Top Level Block Diagram

## Hardware

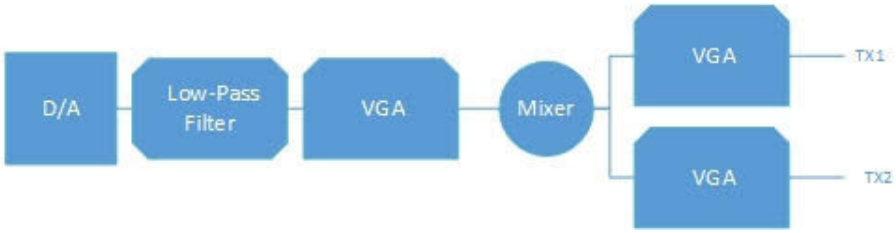


Figure 2: Hardware Block Diagram

## Source Dependencies

- assets/hdl/devices/lime_tx_proxy.rcc/lime_tx_proxy.cc

# Component Spec Properties

| Name | Type | Sequence Length | Array Dimensions | Accessibility | Valid Range | Default | Usage |
|---|---|---|---|---|---|---|---|
| lpf_bw_hz | Float | - | - | Writable, Readable | - | - | The low pass filter that is used to filter out any noise on the received signal. |
| post_lpf_gain_db | Short | - | - | Writable, Readable | - | - | The gain value for the VGA in after the low pass filter. The value is in dB and can only be set in multiples of 3. |
| pre_mixer_dc_offset_i | UChar | - | - | Writable, Readable | - | - | The register value used to tune the DC offset of the transmitted signal to close to zero on the I path. |
| pre_mixer_dc_offset_q | UChar | - | - | Writable, Readable | - | - | The register value used to tune the DC offset of the transmitted signal to close to zero on the Q path. |
| center_freq_hz | Double | - | - | Writable, Readable | - | - | The value of the tuned center frequency of the transmitter. |
| output_gain_db | Short | - | - | Writable, Readable | - | - | The gain value for the VGA in after the mixer. |
| noutputs | UChar | - | - | Readable, Parameter | 1-2 | 1 | The number of hardware outputs that are available to this TX interface. |
| output_select | UChar | - | - | Writable, Readable | - | - | This is the hardware selection of which output to pass the mixer output of the mixer to. The two outputs are identical, but on a given platform, there could be different analog hardware connected outside of the LMS6002D. |

# Worker Properties

## lime_tx_proxy.rcc

| Type | Name | Type | Sequence Length | Array Dimensions | Accessibility/ Advanced | Valid Range | Default | Usage |
|---|---|---|---|---|---|---|---|---|
| SpecProperty | lpf_bw_hz | - | - | - | WriteSync | 14e6, 10e6, 7e6, 6e6, 5e6, 4.375e6, 3.5e6, 3e6, 2.75e6, 2.5e6, 1.92e6, 1.5e6, 1.375e6, 1.25e6, 0.875e6, 0.75e6 | - | The low pass filter that is used to filter out any noise on the received signal. |
| SpecProperty | post_lpf_gain_db | - | - | - | WriteSync | -4 to -35 | - | The gain value for the VGA in after the low pass filter. The value is in dB and can only be set in multiples of 3. |
| SpecProperty | pre_mixer_dc_offset_i | - | - | - | WriteSync | 0x00-0x80 | - | The register value used to tune the DC offset of the transmitted signal to close to zero on the I path. |
| SpecProperty | pre_mixer_dc_offset_q | - | - | - | WriteSync | 0x00-0x80 | - | The register value used to tune the DC offset of the transmitted signal to close to zero on the Q path. |
| SpecProperty | center_freq_hz | - | - | - | WriteSync | 232,500 - 3,720,000 | - | The value of the tuned center frequency of the transmitter. |
| SpecProperty | output_gain_db | - | - | - | WriteSync | 0-25 | - | The gain value for the VGA in after the mixer. |
| SpecProperty | noutputs | - | - | - | - | 2 | 2 | The number of hardware outputs that are available to this TX interface. |
| SpecProperty | output_select | - | - | - | WriteSync | 1-2 | - | This is the hardware selection of which output to pass the mixer output of the mixer to. The two outputs are identical, but on a given platform, there could be different analog hardware connected outside of the LMS6002D. |

# Performance and Resource Utilization

**lime_tx_proxy.rcc**

| Processor Type | Processor Frequency | Run Function Time |
|---|---|---|
| TBD | TBD | TBD |

# Test and Verification

***Note: A component unit test does not exist. Reference the applications/ for a hardware-in-the-loop test application.***

The testbench for this proxy is meant to exercise the properties of the proxy worker dynamically while the application is running. Random data is sent out to emulate noise on the spectrum analyzer. The following steps are taken in the testbench:

1) Change the `output_gain_db` settings
2) Change the `post_lpf_gain_db` settings
3) Toggle the `output_select` value
4) Change the filtering settings.

These steps are repeated at different center frequencies and directions are provided in the testbench for this. The results are inspected visually on a spectrum analyzer. The results should be as follows:

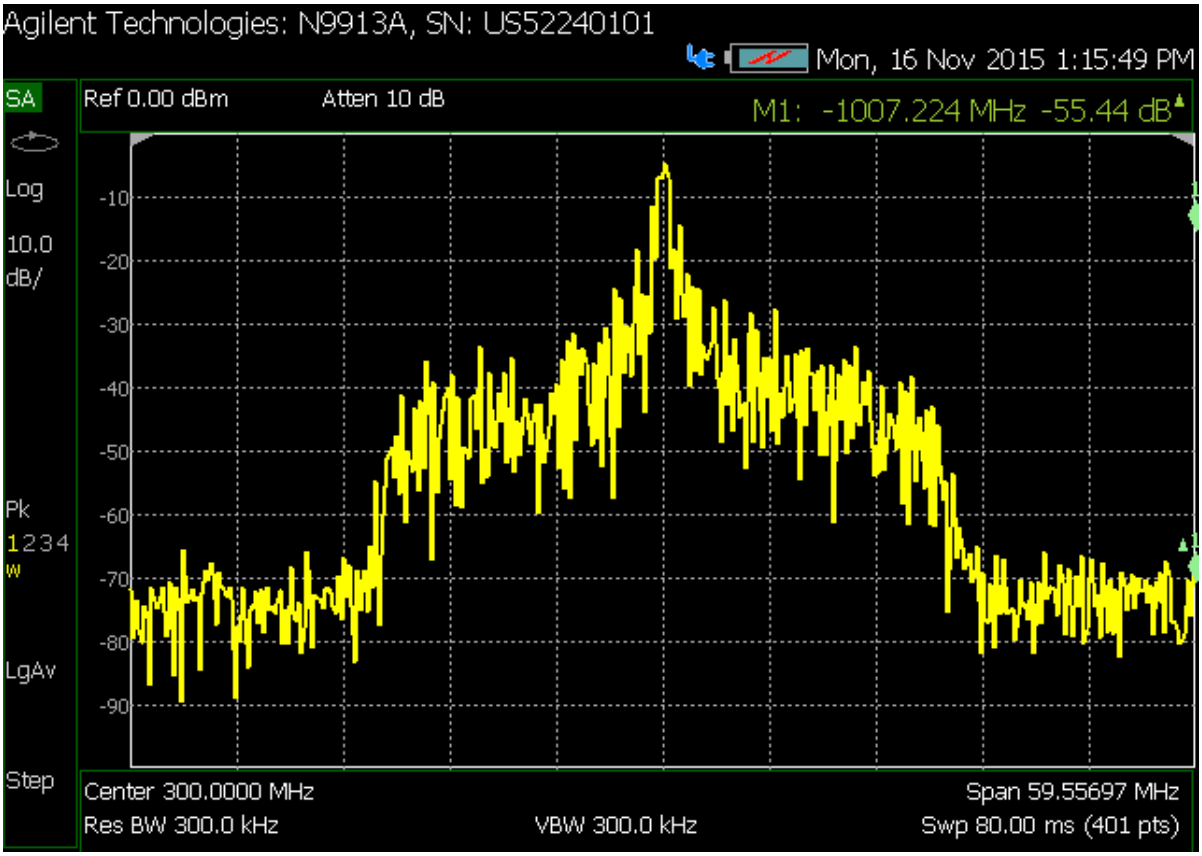


Figure 3: Unit Test Filtered for Matchstiq-Z1

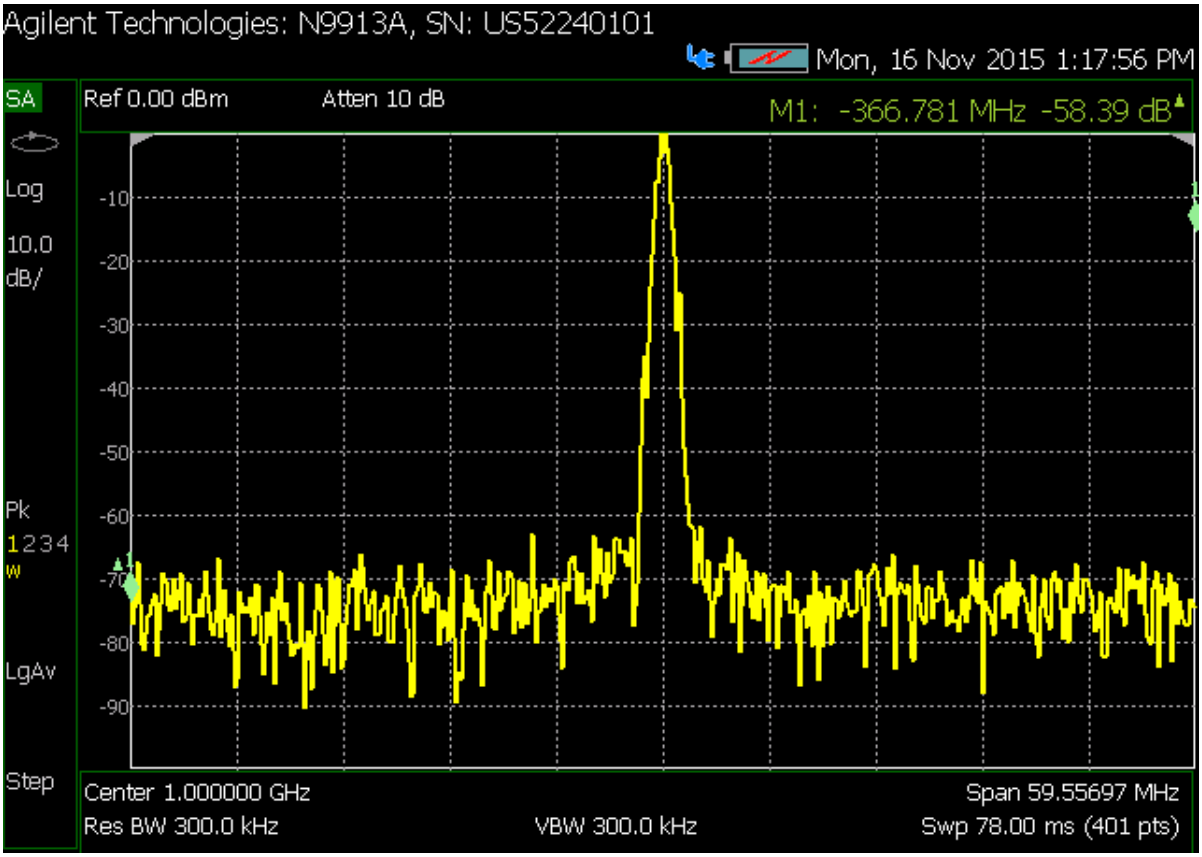


Figure 4: Unit Test Filtered for Matchstiq-Z1

The signal should look like a amplitude adjusted version of the first picture throughout except when testing the filtering. In the filtering stage of testing, the signal will slowly walk from the second picture back to the first picture.

The spectrum for the Zed/Zipper, below, is a result of the Zed/Zipper having a higher minimum sample rate of 500kHz vs 100kHz for the Matchstiq-Z1. In this case, the Zynq processor can not source sample data fast enough, which results in the output buffer underrunning. During these underrun conditions, the output of the buffer holds the last value, which results in repeated constant values being sent to the Lime transmitter.

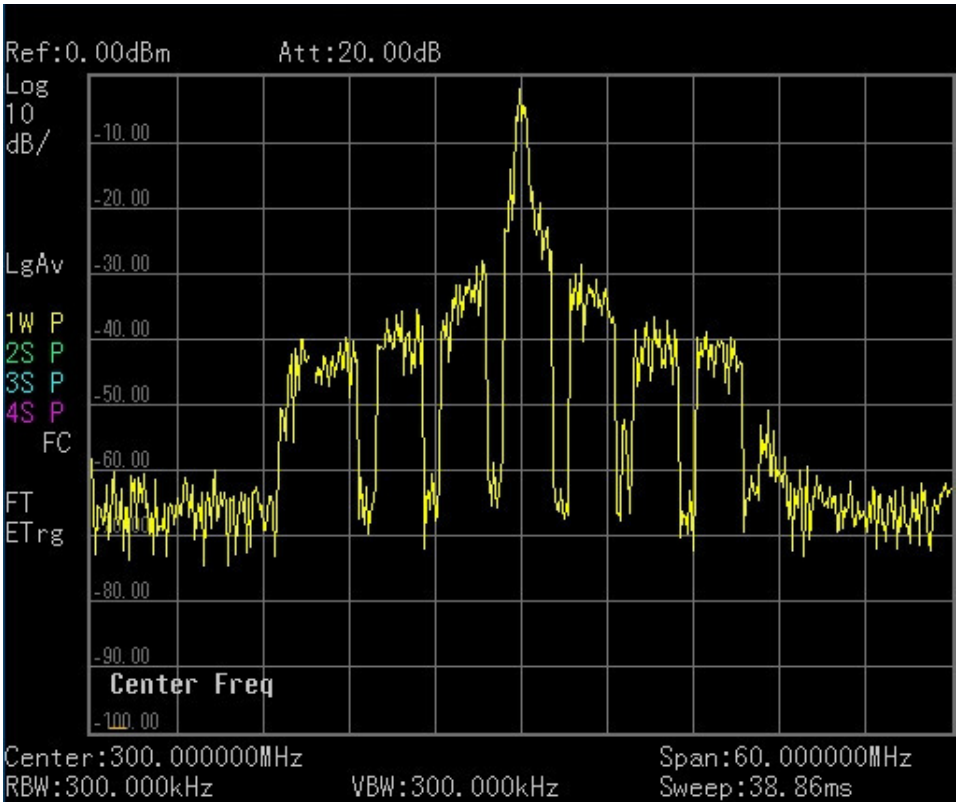


Figure 5: Unit Test Filtered for Zed/Zipper

# References

1) LMS6002D Datasheet, www.limemicro.com