

Bitshark FMC-1RX Reference Design User's Manual for the Xilinx SP605/ML605

Version 1.1



Disclaimer

Epiq Solutions is disclosing this document (“Documentation”) as a general guideline for development. Epiq Solutions expressly disclaims any liability arising out of your use of the Documentation. Epiq Solutions reserves the right, at its sole discretion, to change the Documentation without notice at any time. Epiq Solutions assumes no obligation to correct any errors contained in the Documentation, or to advise you of any corrections or updates. Epiq Solutions expressly disclaims any liability in connection with technical support or assistance that may be provided to you in connection with the Information.

THE DOCUMENTATION IS DISCLOSED TO YOU “AS IS” WITH NO WARRANTY OF ANY KIND. EPIQ SOLUTIONS MAKES NO OTHER WARRANTIES, WHETHER EXPRESSED, IMPLIED, OR STATUTORY, REGARDING THE DOCUMENTATION, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT WILL EPIQ SOLUTIONS BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR INCIDENTAL DAMAGES, INCLUDING ANY LOSS OF DATA OR LOST PROFITS, ARISING FROM YOUR USE OF THE DOCUMENTATION.

All material in this document is Copyrighted by Epiq Solutions 2011. All trademarks are property of their respective owners.

Table of Contents

1	About this Document.....	5
2	Legal Considerations.....	5
3	Proper Care and Handling.....	5
4	References.....	6
5	Overview.....	7
	5.1 Hardware/Software Requirements.....	8
	5.2 Setting up the Hardware.....	8
	5.3 Running the FMCGU Spectrum Analyzer Application.....	11
6	Configuring and Using the FMCGU.....	14
	6.1 Time Domain Parameters.....	15
	6.2 Frequency Domain Parameters.....	15
	6.3 Control Panel Parameters.....	16
	6.4 FPGA Reference Design Command Protocol Specification.....	17
	6.5 Host FPGA I/Q Sample Acquisition.....	19
7	FPGA Logic.....	21
	7.1 FPGA Top Level.....	21
	7.2 FPGA ADC Interface.....	21
	7.3 FPGA Signal Conditioning.....	21
	7.4 FPGA UART Interfaces.....	22
	7.5 FPGA Frame FIFO.....	22
8	FMC-1RX Command Protocol Specification.....	23
	8.1 Overview.....	23
	8.2 Command Specification.....	24
	8.3 Parameter Specification.....	25

Revision History

Date	Revision	Description
0.1	12/20/2010	Initial pre-release
0.2	01/10/2011	Update to finish most of the basic procedures for using the reference design
0.3	01/28/2011	Update FPGA Logic section
1.0	01/20/2012	Minor updates, as needed for Rev C of the FMC-1RX card
1.1	02/15/2012	Update Register A1 default value for Rev C

1 About this Document

This document provides the necessary details for using the spectrum analyzer reference design provided with the Bitshark™ FMC-1RX RF receiver card [1] developed by Epiq Solutions. This reference design consists of an FPGA bitstream that is loaded into the Xilinx SP605 or ML605 FPGA evaluation board (purchased separately through, as well as a PC-based spectrum analyzer application that displays time/frequency domain of the received RF spectrum.

2 Legal Considerations

The Bitshark FMC-1RX card is distributed all over the world. Each country has its own laws governing reception of radio frequencies. The user of the Bitshark FMC-1RX card and associated software is solely responsible for insuring that it is used in a manner consistent with the laws of the jurisdiction in which it is used. Many countries, including the United States, prohibit the reception of certain frequency bands, or receiving certain transmissions without proper authorization. Again, the user is solely responsible for the user's own actions.

3 Proper Care and Handling

The Bitshark FMC-1RX card is fully tested by Epiq Solutions before shipment, and is guaranteed functional at the time it is received by the customer, and **ONLY AT THAT TIME**. Improper use of the FMC-1RX can easily cause it to become non-functional, so Epiq Solutions cannot make any guarantees as to its function over time. Additionally, the use of the FMC-1RX in conjunction with any FPGA host platform (such as the Xilinx SP605/ML605) poses the risk that damage may be done to the host platform due to improper use. In particular, a list of actions that are very likely to cause damage to the card and/or host platform include the following:

- Adding or removing the card to a host system while the host system is powered up
- Allowing metal objects to touch the circuit board (on either side) while it is powered up
- Connecting a transmitter into the FMC-1RX card without proper attenuation
- Handling the FMC-1RX card without proper static precautions (ESD protection)
- Connecting a cable to either the external clock out or external clock in signal ports while the FMC-1RX is powered up
- Executing an FPGA bitstream in the host FPGA platform that was developed by an end-user

The above list is not comprehensive, and experience with the appropriate measures for handling unboxed electronic devices is required. Lastly, usage and handling instructions for the Xilinx SP605/ML605 can be found in the documentation section of [3] and [4].

4 References

- [1] Epiq Solutions Website:
<http://epiqsolutions.com/>

- [2] Bitshark FMC-1RX Product Page:
<http://epiqsolutions.com/fmc-1rx.php>

- [3] Xilinx ML605 product page:
<http://www.xilinx.com/products/devkits/EK-V6-ML605-G.htm>

- [4] Xilinx SP605 product page:
<http://www.xilinx.com/products/devkits/EK-S6-SP605-G.htm>

- [5] VITA 57 Specification:
<http://www.vita.com/specifications.html>

- [6] Silicon Labs website for installing the CP2101 USB-to-UART device driver:
<http://www.silabs.com/products/mcu/pages/usbtouartbridgevcpcdrivers.aspx>

- [7] Linear Technology LTC2267 A/D converter datasheet:
<http://cds.linear.com/docs/Datasheet/22687614fa.pdf>

- [8] Ellingson, S.W. "Correcting I-Q Imbalance in Direct Conversion Receivers " 2003
http://argus.naapo.org/~rchilders/swe_argus_pubs/iqbal.pdf

- [9] Xilinx Application Notes (XAPP866, XAPP1064, XAPP878, XAPP879)
http://www.xilinx.com/support/documentation/application_notes.htm

5 Overview

The Bitshark FMC-1RX card is a broadband, high performance, single tuner RF receiver card that is compliant with the FPGA Mezzanine Card (FMC/VITA 57) digital interface standard [5]. It supports a tuning range between 300 MHz and 4 GHz, while supporting user-defined channel bandwidths up to 50 MHz. Digitized baseband I/Q samples are provided by dual A/D converters, and are accessible on the FMC interface for use by the host system. The FMC-1RX card is targeted at commercial wireless radio applications, supporting a variety of common frequency bands. Additional details of the FMC-1RX card can be found in [2].

In order to evaluate the Bitshark FMC-1RX, a reference design has been developed to allow an end-user to combine the FMC-1RX with a Xilinx FPGA evaluation board and a host PC to provide a basic spectrum analyzer. This spectrum analyzer provides graphical controls to allow the various parameters on the FMC-1RX card (RF center frequency, RF gain, etc.) to be modified. Once the FMC-1RX card is configured, contiguous blocks of digitized baseband I/Q samples are transported up to the PC over a USB 2.0 interface. The PC then displays the time and frequency representation of the received data. A block diagram of the reference design system is shown in Figure 1.

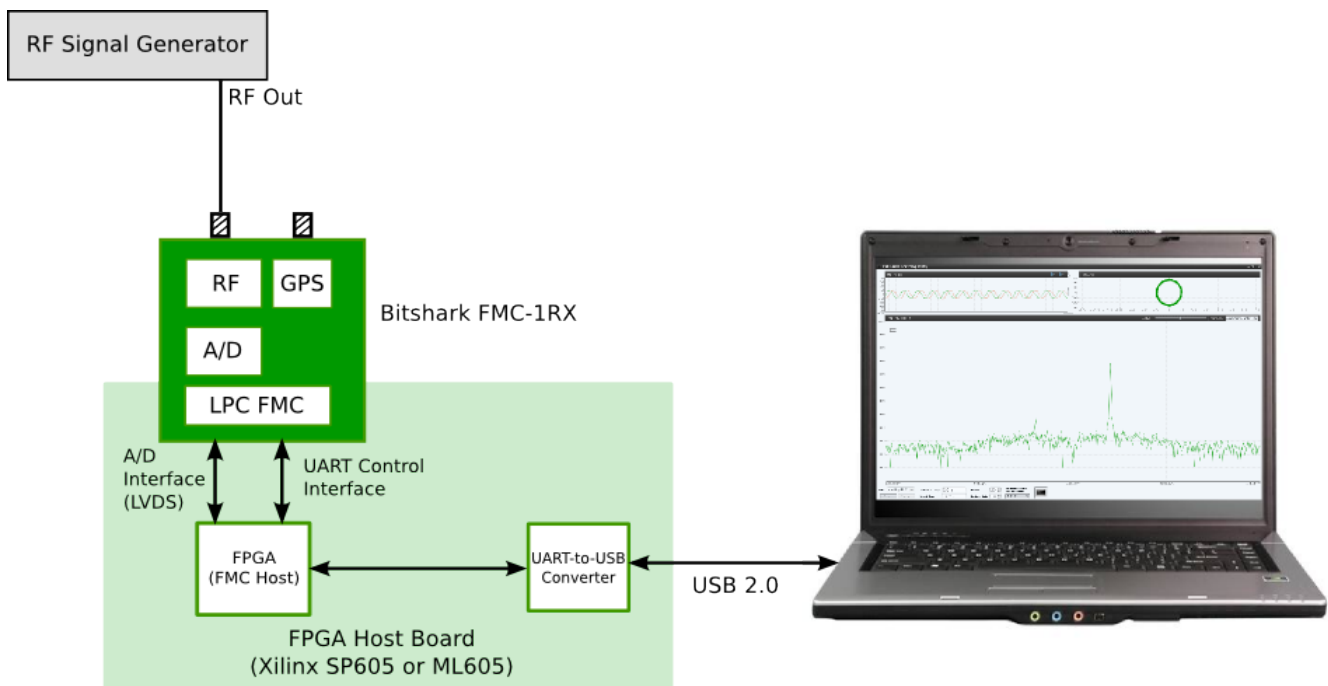


Figure 1: Block diagram of the FMC-1RX reference design system

This reference design supports both the Xilinx ML605 (which uses the Virtex® 6 XC6VLX240T FPGA) and the SP605 (which uses the Spartan® 6 XC6SLX45T). The pre-compiled bitstreams for each platform can be found in the “Downloads” section of [2]. For those users who purchased the Bitshark FMC-1RX-DVK, please contact Epiq Solutions for access to the HDL source code for both of these platforms.

The PC-based spectrum analyzer application is written in Java and has been tested under both Windows (XP, Vista, and 7) and Linux (Fedora 11, 12, and 13). Other versions of Windows and Linux may also work but are untested and up to the end user to verify functionality.

5.1 Hardware/Software Requirements

In order to execute and validate the FMC-1RX reference design, the components listed in the table below are required:

Name	Description
FPGA host platform (either the Xilinx ML605 or SP605)	FMC host board for the Bitshark FMC-1RX card
Bitshark FMC-1RX	RF receiver card in an FMC form-factor
RF signal generator	Generates an RF input signal for the FMC-1RX to receive
SMA cable	Used to connect the RF signal generator to the FMC-1RX card
USB A to mini B cable	Used to connect the FPGA host platform to the PC
PC (Windows or Linux-based)	Executes the Java-based spectrum analyzer application
Silicon Labs CP2101 USB-to-UART device driver	Device driver software that needs to be installed on the PC to allow the USB-to-UART converter on the FPGA host platform to be recognized as a serial/COM port.
FMC Graphing Utility (FMCGU)	Java-based spectrum analyzer software application that executes on the PC (includes all necessary Java runtime environment requirements and libraries, removing any external dependencies on the host PC).

5.2 Setting up the Hardware

The following section outlines the steps required to assemble all of the hardware utilized by the reference design for the FMC-1RX card. Note: In several locations, reference is made to the “FPGA host platform”. This is referring to either the ML605 or SP605, and applies equally to both. In cases where there are specific instructions related to one of these two boards, it is explicitly called out.

5.2.1 Step 1: Verify that the FPGA host platform is functioning properly.

The FPGA host platform has a series of validation steps that can be executed to ensure that it is properly functioning independent of the FMC-1RX card. For the ML605, these steps can be found in [3]. For the SP605, these steps can be found in [4].

5.2.2 Step 2: Install the CP2101 USB-to-UART device driver on the PC

In order for either the SP605 or the ML605 to be able to communicate with the host PC through the on-board

USB-to-UART converter chip, the appropriate device driver needs to be installed on the host PC. Directions for installing this device driver can be found in the appropriate user's manual for [3] and [4]. Alternately, directions for installing the CP2101 device driver can be found at the Silicon Labs website [6].

5.2.3 Step 3: Connect the FMC-1RX card to the host FPGA platform

With the FPGA host platform powered down, insert the FMC-1RX into the low-pin count (LPC) FMC slot on the FPGA host platform. Ensure that the FMC-1RX card is fully seated in the FMC slot. Figure 2 shows the FMC-1RX card in the ML605. Figure 3 shows the FMC-1RX card in the SP605.

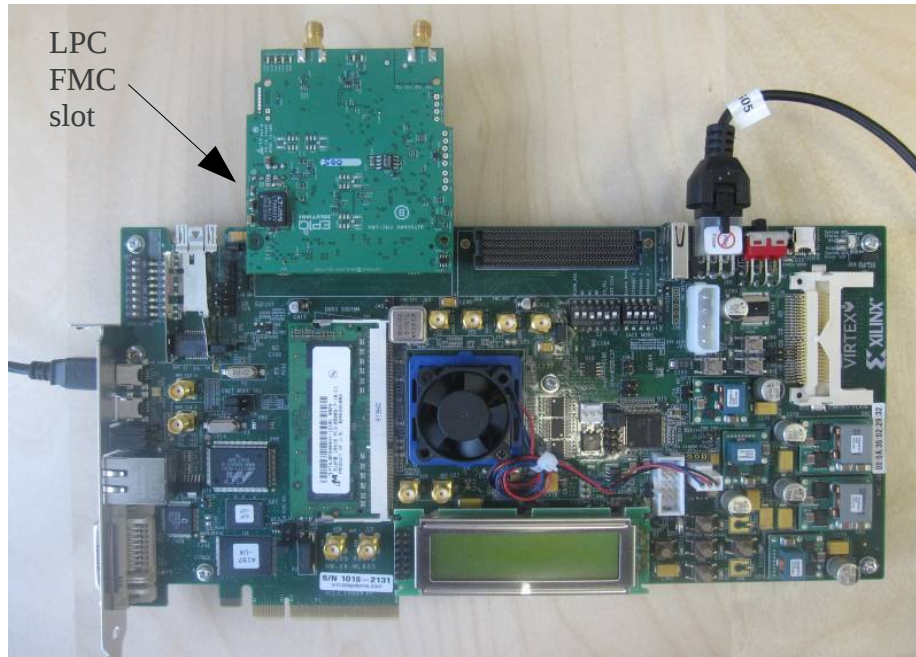


Figure 2: Xilinx ML605 with the FMC-1RX card inserted in the LPC slot



Figure 3: Xilinx SP605 with the FMC-1RX card inserted in the LPC slot

5.2.4 Step 3: Connect the RF signal generator to the FMC-1RX card

In order to verify the FMC-1RX card is functioning properly, it is necessary to connect the output of an RF signal generator to the RF input of the FMC-1RX card. For testing purposes, set the output level of the RF signal generator to -80 dBm at a frequency of 2001 MHz. Then connect the RF output from the signal generator to the “RF1-IN” SMA input of the FMC-1RX card. Figure X shows this connection.

<TODO: add picture of FMC card with SMA cable plugged into RF1-IN input>

5.2.5 Step 4: Configure the FPGA host platform with the reference design bitstream.

Both the SP605 and the ML605 can be programmed over JTAG using Xilinx's iMPACT programming tool. The steps involved in programming either FPGA board can be found in [3] and [4]. The latest reference design bitstream can be found at [2]. Power up the FPGA host platform, and program the FPGA with the reference design bitstream.

5.2.6 Step 5: Connect the FPGA host platform to the PC.

The PC that will run the FMCGU spectrum analyzer application connects to the FPGA host platform through a USB A to mini USB cable. Connect the USB A end of the cable to an available USB port on the PC, and the mini USB end of the cable to the mini USB port labeled “UART” on the FPGA host platform, as shown in Figure 4.

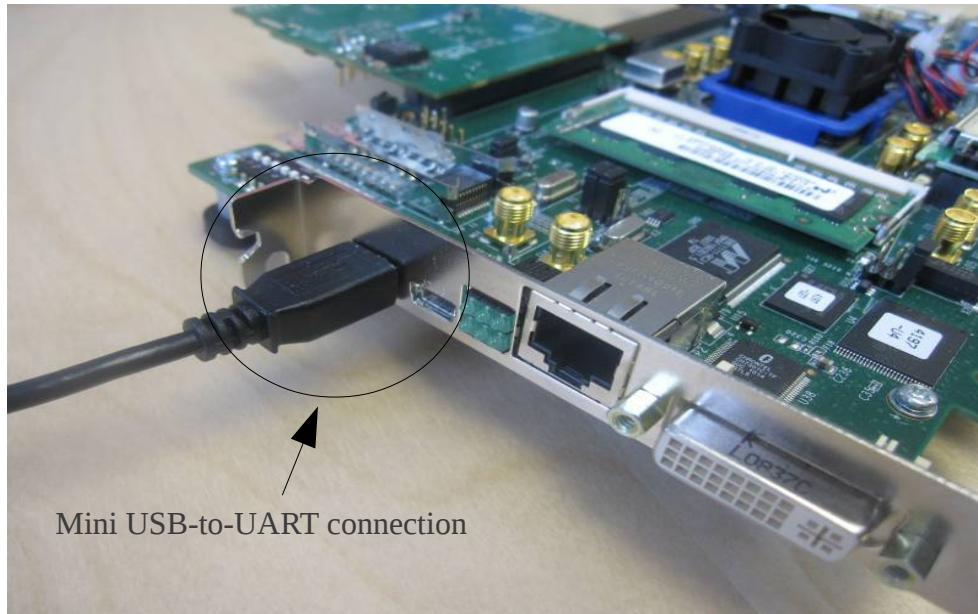


Figure 4: UART-to-USB serial connection on host FPGA platform

5.3 Running the FMCGU Spectrum Analyzer Application

The following section outlines the steps needed to run the FMCGU spectrum analyzer application on the PC. Note that the FMCGU application runs under multiple operating systems. Where appropriate, separate directions are provided for each operating system.

5.3.1 Step 6: Download and install the FMCGU application

The FMCGU application can be downloaded from [2]. The zipfiles/tarballs available for download contain the application itself, as well as the Java runtime environment necessary to execute the application. This application has no external dependencies, and includes all the necessary libraries for execution. Once the application zipfile/tarball has been downloaded, decompress the downloaded file to the directory where the application will reside. Note: There are both 32-bit and 64-bit installations for Linux, as well as a Windows version that has been tested on XP, Vista, and Windows 7.

5.3.2 Step 7: Launch the FMCGU application

After decompressing, the resultant FMCGU directory will contain a script for launching the FMCGU application.

- Windows users:
 - Open a DOS shell by left-clicking on Start->Programs->Accessories->Command Prompt
 - Change directories to where the FMCGU application was decompressed
 - Run the batch file called window_run.bat, which will launch the FMCGU user interface:
 - `C:\FMCGU> window_run.bat`
- Linux users:
 - Open a console shell.
 - Change directories to where the FMCGU application was decompressed
 - As root (either through su or sudo), execute the appropriate shell script, which will launch the FMCGU user interface:
 - 32-bit Linux: `# sudo ./linux32_run.sh`
 - 64-bit Linux: `# sudo ./linux64_run.sh`

The screenshot shown in Figure 5 will be displayed when the FMCGU application launches:

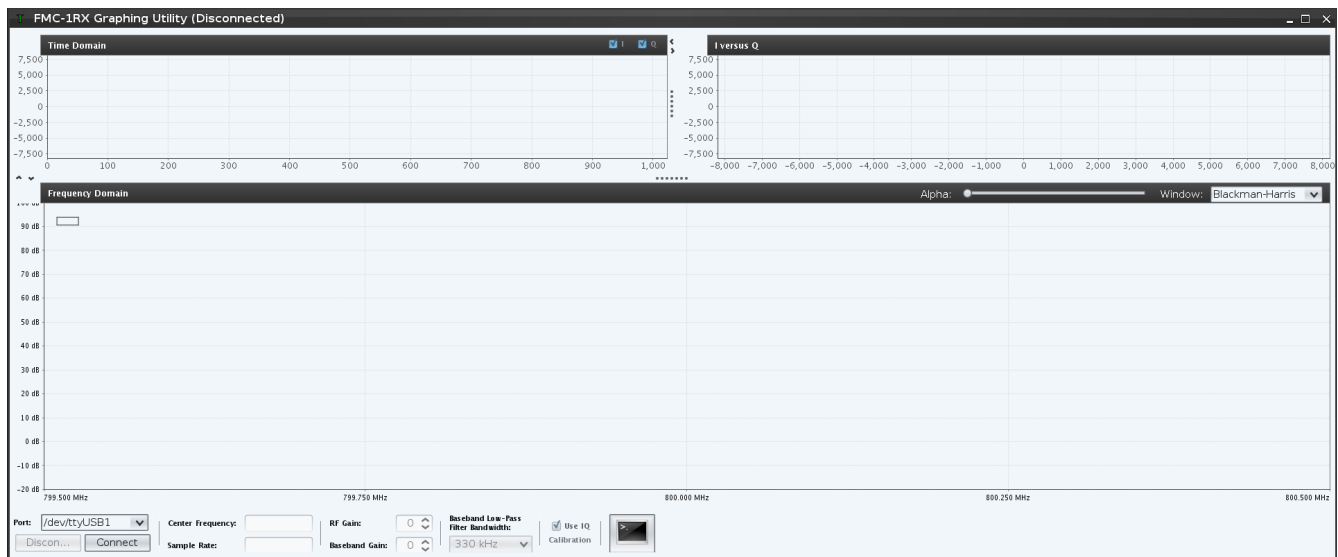


Figure 5: Opening screenshot of FMCGU

5.3.3 Step 8: Connect the FMCGU application to the ML605

Connect the FMCGU application to the ML605 by selecting the appropriate name of the serial port registered to the ML605 when it is plugged in, as shown in Figure 6.

- Windows users
 - Navigate to the Device Manager (Start->Control Panel->System->Device Manager). Expand the “COM Ports” line. A COM port should be assigned to the CP2101 USB-to-UART device on the

ML605. This is the COM port to use in the FMCGU.

- Linux users
 - The device node assigned to the CP2101 USB-to-UART converter on the ML605 can be found by running the `dmesg` command after the ML605 USB cable is plugged into the PC. An output similar to the following should result from `dmesg`:

```
usb 1-1.3: new full speed USB device using ehci_hcd and address 22
usb 1-1.3: New USB device found, idVendor=10c4, idProduct=ea60
usb 1-1.3: New USB device strings: Mfr=1, Product=2, SerialNumber=3
usb 1-1.3: Product: CP2103 USB to UART Bridge Controller
usb 1-1.3: Manufacturer: Silicon Labs
usb 1-1.3: SerialNumber: 0001
usb 1-1.3: configuration #1 chosen from 1 choice
cp2101 1-1.3:1.0: cp2101 converter detected
usb 1-1.3: reset full speed USB device using ehci_hcd and address 22
usb 1-1.3: cp2101 converter now attached to tttyUSB1
```

Select the appropriate serial port device name, and left-click on the “Connect” button.

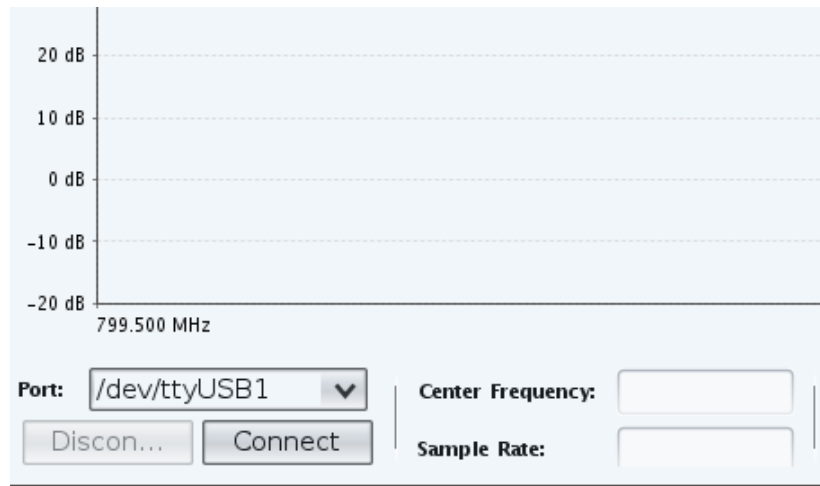


Figure 6: FMCGU screenshot showing the serial port connection options

Once connected, log messages will be displayed to the console where the application was launched. These messages contain a log of all the commands sent to the ML605 reference design to configure the FMC-1RX card, as well as a variety of additional status messages. Once the initial configuration is complete, the different plot windows in the FMCGU will be updated with the baseband I/Q sample data received from the ML605. For the default case, the FMC-1RX tunes to an RF frequency of 2000 MHz. With an RF signal generator connected to the FMC-1RX, and providing a -80 dBm tone at 2001 MHz, the screenshot shown in Figure 7 should be seen.



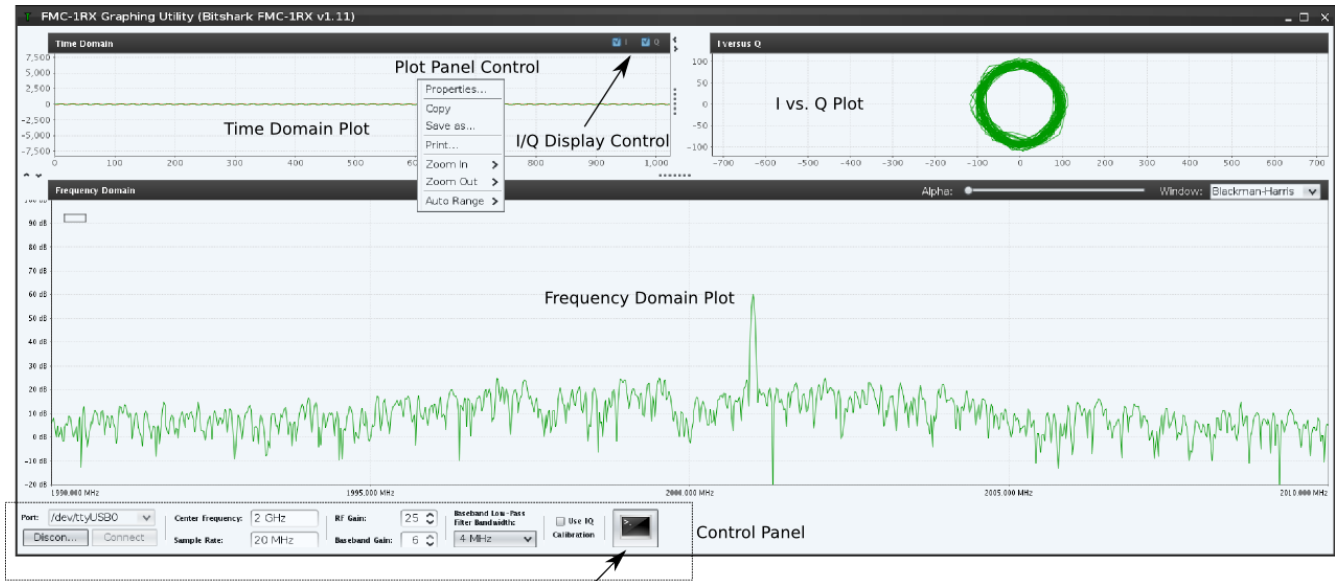
Figure 7: FMCGU screenshot showing an RF signal with the default configuration

6 Configuring and Using the FMCGU

The FMCGU is divided up into four main user interface elements:

- Time Domain plot-This provides an amplitude versus time plot of the signal that is received on both the I and Q channels. The amplitude axis displays the raw two's complement signed 14-bit values provided by the A/D converters.
- Frequency Domain plot-This provides an amplitude versus frequency plot of the signal that is received. The amplitude axis is logarithmic, in dB (not absolute dBm), and the frequency axis is linear in MHz.
- The I versus Q plot-This provides a plot of the I channel data versus the Q channel data. The axis are measured in the same amplitude units as the Time Domain plot.
- Control Panel-This provides a series of controls to allow the user to modify the configuration of the FMC-1RX card in real-time.

A screenshot with all enumerated controls called out is shown in Figure 8. The following section provides an overview over the different configurable parameters in each of the different user interface elements in the FMCGU.



FPGA Host Platform Console
 Figure 8: FMCGU Screenshot with enumerated displays

6.1 Time Domain Parameters

6.1.1 I/Q Display Control

The I/Q Display Control defines which time domain channels are displayed. The “I” channel and “Q” channel can be turned on or off by selecting the appropriate checkbox.

6.1.2 Plot Panel Control

The Plot Panel control provides various capabilities related to the selected plot window, including the ability to zoom in/out, and options for saving the plot screen output. This control is accessed by right-clicking anywhere on the Time Domain plot window.

6.2 Frequency Domain Parameters

6.2.1 Alpha

The Alpha parameter controls how much averaging is performed on the received signal. Increasing the averaging helps to reduce the visual affects of the noise on the received signal, allowing weak continuous signals to show up above the noise floor. However, increasing the averaging also increases the delay in seeing the effects of a rapidly changing signal.

6.2.2 Window

The Window parameter provides a list of different windowing operations that can be performed on the data prior to displaying the frequency domain representation. This helps to reduce leakage of signal energy to adjacent FFT bins by shaping the incoming time domain data. The default window used is the Blackman-Harris window, though a number of different windowing options are available. A rectangular window can be selected to disable the windowing operation.

6.2.3 Plot Panel Control

The Plot Panel control provides various capabilities related to the selected plot window, including the ability to zoom in/out, and options for saving the plot screen output. This control is accessed by right-clicking anywhere on the Frequency Domain plot window.

6.3 Control Panel Parameters

6.3.1 Center Frequency

The Center Frequency parameter controls the RF tuning frequency. The range of this parameter is from 300 MHz to 4 GHz. The frequency can be entered in units of MHz or GHz.

6.3.2 Sample Rate

The Sample Rate parameter control the sample rate of the dual A/D converters sampling the I and Q data channels. The range of this parameter is from 5 MHz to 105 MHz.

6.3.3 RF Gain

The RF Gain parameter control the RF gain of the front-end. The range of this gain is from 0 dB to 31.5 dB, in 0.5 dB steps.

6.3.4 Baseband Gain

The Baseband Gain parameter controls the baseband gain. The range of this gain is from 0 dB to 6 dB, in 3 dB steps.

6.3.5 Baseband Low-Pass Filter Bandwidth

The Baseband Low-Pass Filter Bandwidth parameter control the 3 dB down frequency of the baseband low-pass filter. This drop-down box provides the list of available filter cutoff frequencies, from 330 KHz to 28 MHz.

6.3.6 Use IQ Calibration

The Use IQ Calibration checkbox controls whether or not the FPGA host platform performs I/Q imbalance correction of the received signal. This relies on calibration values stored in the FMC-1RX card, which is calibrated over frequency at the time of manufacture.

6.3.7 FPGA Host Platform Console

The FPGA Host Platform Console provide a mechanism to interact with the FPGA host platform through the low-level serial interface utilized by the FMCGU. This allows an advanced user to access all parameters provided by both the FPGA reference design as well as the FMC-1RX card. For additional details of this serial interface and command protocol specification, see Section 6.4.

6.4 FPGA Reference Design Command Protocol Specification

The FPGA host board utilizes a UART as the primary interface to the system. This UART interface provides the following functionality:

- Read/Write of configuration registers located in the host FPGA
- Read/Write FMC-1RX commands that are passed through to the FMC-1RX card
- Read fixed-length blocks of contiguous I/Q samples received from the FMC-1RX card

The high-level requirements of the FPGA host UART interface are listed below:

- The UART interface operates at 115200 baud, with 8 data bits, 1 stop bit, no parity, and no flow control.
- Commands are sent to and received from the FPGA host platform as simple human-readable ASCII text.
- The ASCII text command interface utilizes all uppercase text. The one exception here is the “grab” command for reading blocks of I/Q samples, which uses lower-case (see below).
- All commands start with a single character routing code. This routing code determines the destination of the command to follow. There are currently two supported routing codes:
 - The 'B' routing code indicates that the command is destined for the Bitshark FMC-1RX card, and should thus be passed through to the FMC-1RX's serial interface without modification.
 - The 'F' routing code indicates that the command is destined for the FPGA host logic.
- The FPGA host immediately echos all received characters back over its UART interface to confirm reception (regardless of routing code).
- All commands must be terminated with either a '\r' (carriage return) character or a '\n' (newline) character.
- Upon reception of either the '\r' or '\n' terminating character, the FPGA host will echo back “\r\n”. This is the only case where the echo is not exactly what is received.
- Responses from the FPGA host consist of one or more lines that are terminated with a “\r\n” character sequence.
- All commands result in a response that contains, at minimum, an ACK or NCK indicating the command was acknowledged or not acknowledged.
- Information messages may be sent by the FPGA host as part of a response (in addition to the actual contents of the response). These messages begin with “I:”, contain non-critical details of the response, and terminate in “\r\n”.
- Error messages may be sent by the FPGA host as part of a response. These messages begin with “E:”, contain an error message and terminate in “\r\n”.
- All commands, regardless of success or failure, result in a '>' character (without a terminating “\r\n”) being sent by the FPGA host as the final character in a response to indicate that it is ready for a new command.

6.4.1 Host FPGA Configuration Registers

The FPGA host provides a series of 32-bit registers that provide control of various parameters in the reference design. The table below provides the details of the register map. Note: Currently, it is only possible to write to the host FPGA configuration registers. Read operation is not supported.

Address: 0

Description: Register A0 of LTC2267 A/D converter

Bit Map:

Bits 31-8: Reserved	Bits 7-0: Register A0 (see [7] for details) Default (set by FMCGU): 0x00000080
---------------------	---

Address: 1

Description: Register A1 of LTC2267 A/D converter

Bit Map:

Bits 31-8: Reserved	Bits 7-0: Register A1 (see [7] for details) Default (set by FMCGU): 0x00000020
---------------------	---

Address: 2

Description: Register A2 of LTC2267 A/D converter

Bit Map:

Bits 31-8: Reserved	Bits 7-0: Register A2 (see [7] for details) Default (set by FMCGU): 0x00000001
---------------------	---

Address: 3

Description: Register A3 of LTC2267 A/D converter

Bit Map:

Bits 31-8: Reserved	Bits 7-0: Register A3 (see [7] for details) Default (set by FMCGU): 0x00000000
---------------------	---

Address: 4

Description: Register A4 of LTC2267 A/D converter

Bit Map:

Bits 31-8: Reserved	Bits 7-0: Register A4 (see [7] for details) Default (set by FMCGU): 0x00000000
---------------------	---

Address: 5

Description: Reserved

Bit Map:

Bits 31-0: Reserved

Address: 6

Description: Reserved

Bit Map:

Bits 31-0: Reserved

Address: 7

Description: Reserved

Bit Map:

Bits 31-0: Reserved

Address: 8

Description: DC offset control (high-pass filter implemented in the FPGA)

Bit Map:

Bits 31-1: Reserved	Bits 0: DC offset correction state 0=Disabled 1=Enabled Default (set by FMCGU): 1 (Enabled)
---------------------	--

Address: 9

Description: I/Q imbalance correction parameters (see [8] for description of algorithm and coefficients)

Bit Map:

Bits 31-26: Reserved	Bits 25-22: I/Q imbalance correction coefficient 'C', integer portion Range: 0-16 Default (set by FMCGU): 0	Bits 21-16: I/Q imbalance correction coefficient 'C', fractional portion Range: 0-63 (multiply value by 1/64 to determine actual fractional contribution) Default (set by FMCGU): 0	Bits 9-6: I/Q imbalance correction coefficient 'D', integer portion Range: 0-16 Default (set by FMCGU): 1	Bits 5-0: I/Q imbalance correction coefficient 'C', fractional portion Range: 0-63 (multiply value by 1/64 to determine actual fractional contribution) Default (set by FMCGU): 0
-------------------------	---	---	---	---

The following section provides an example usage scenarios for updating registers in the FPGA host register map.

Scenario: A user wants to update the contents of register 1 on the LTC2267:

Command Sequence:

```
> F W 01 0000002C
```

```
ACK
```

```
>
```

Scenario: A user wants to disable DC offset.

Command Sequence:

```
> F W 08 00000000
```

```
ACK
```

```
>
```

TODO: Update to indicate how negative #s work for 'C' and 'D'

6.5 Host FPGA I/Q Sample Acquisition

The FPGA host reference design supports the ability to acquire a contiguous block of samples at the current sample rate (up to 105 Msamples/sec), store them to Block RAM internal to the FPGA, and transfer them up to the PC over the UART interface. Currently, the only block size supported for this transfer is 1K samples (1024 I/Q pairs), but additional block sizes can be supported through modifications to the source code of the reference design.

This command is called the “grab” command, with the simple mnemonic 'g'. This command is slightly different

than the other 'F' commands. In addition to being the only command that uses a lowercase letter, it also contains 16-bit binary values for the 'I' and 'Q' data returned (which are not typically human readable). An example acquisition is shown in the following scenario:

```
>F g
ACK\r\n
<Sample 0, 16-bit 'I'><Sample 0, 16-bit 'Q'><Sample 1, 16-bit 'I'>< Sample 1, 16-bit 'Q'>...<Sample 1023, 16-bit 'I'><Sample 1023, 16-bit 'Q'>>
```

Note: The 16-bit binary values are represented above with '<' and '>' characters around them. These do not appear in the actual datastream. Also note that the '>' character is still inserted at the end of the response to indicate the response is complete.

Each 16-bit value consists of a little endian two's complement, sign-extended 14-bit value (due to the fact that the A/D converters are only 14-bits).

7 FPGA Logic

This section provides an overview of the functionality provided by the host FPGA in the reference design. The Verilog source code for this reference design is provided with the Bitshark FMC-1RX-DVK. Contact Epiq Solutions for additional details.

The FPGA reference design is available for the Xilinx SP605 and the ML605. While the basic functionality is the same, there are of course differences in the designs. The main blocks present in both of the designs are:

- Top Level (including frame control, clock generation, and other logic)
- Signal Conditioning (including DC offset correction and I/Q imbalance correction)
- ADC Interface (including high-speed serial data, and SPI bus control)
- UART Interfaces (USB and Bitshark FMC-1RX uC)
- Frame FIFO

7.1 FPGA Top Level

The Top Level module instantiates the other modules listed above, as well as provide various other functions to make the design work properly. Control logic for the framing of the I/Q samples is included at the top level. In addition, the two UART and the ADC SPI clocks are created here.

The inputs and outputs define the I/O used on the SP605 / ML605, including oscillator(s), buttons, LEDs, USB UART pins, and FMC pins. These input and output names must match those defined in the .ucf file.

Several configurable parameters are defined at the top level as well.

7.2 FPGA ADC Interface

The ADC Interface block contains the logic required to take in high-speed serial data (i.e. I/Q samples), as well as a SPI bus master for configuring the ADC chip. This block contains the most differences between the SP605 and ML605 due to the unique clocking and I/O primitives available in the Spartan-6 and Virtex-6 devices. The serial interface is defined by the ADC chip and how it is configured. For details and timing diagrams, see the datasheet [7].

The FPGA logic of the high-speed serial interface uses several Xilinx I/O and clock primitives, including SERDES. In addition, both the SP605 and the ML605 designs leveraged existing reference designs / application notes available from Xilinx, including: XAPP866, XAPP1064, XAPP878, and XAPP879. Please refer to these application notes while looking at the Verilog code [9].

7.3 FPGA Signal Conditioning

Two modules are included to perform signal conditioning on the I/Q samples prior to being sent out. Additional blocks could be added here, as required by the user's end application.

7.3.1 FPGA DC Offset Correction

The first signal conditioning block is a DC offset detection and correction block consisting of a single-pole IIR low pass filter. Input data is passed through this IIR filter to create a long term average of the DC component of the signal. This DC component is then subtracted from all incoming data. This offset may vary both between I and Q and slowly over time. The IIR filter is a single-pole filter where the pole is real and placed very near the unit circle (at 0.999 for example), which creates a low pass filter with a very narrow passband around DC.

The DC offset correction block has an active-high enable bit; when low, the block is effectively disabled and the input samples are passed through unmodified.

7.3.2 FPGA I/Q Imbalance Correction

The second signal conditioning block is I/Q imbalance correction, which adjusts the phase and amplitude of Q to account for phase and amplitude differences between the I and Q paths during downconversion. The coefficients C and D are configurable, and their calculation is described in [8]. Phase and amplitude errors increase the size of the negative sideband, which introduces a frequency-flipped version of the desired signal at baseband.

The I/Q balancing block has an active-high enable bit; alternatively, when the coefficient values are set such that $C = 0.0$ and $D = 1.0$, the block is also effectively disabled and the input samples are passed through unmodified.

Updates to the coefficient values can be passed down from software as shown in Section 8.3.6. The values are 10 bits each, where the four most significant bits represent the integer portion, and the six least significant bits represent the decimal portion. So, as mentioned above, to simply pass the input samples through without modification, the C coefficient input would be 0x000 (decimal 0.0), and the D coefficient input would be 0x040 (decimal 1.0).

7.4 FPGA UART Interfaces

The FPGA communicates with both the FMC-1RX and the on-board USB-UART converter using a UART interface. Control messages come down from the PC via USB and get converted to UART transactions for the FPGA to respond to. Several registers are available in the FPGA and are listed in Section 6.4.1. Some messages are directed toward the uC on the FMC-1RX, in which case the FPGA acts as a pass-thru. Because the UARTs are configured to operate at different baud rates, the FPGA contains logic to handle this (in both directions).

In addition to control messages, frames of samples are sent up to the PC over the same UART interface (converted to USB transactions by the on-board USB-UART converter, as mentioned previously). The frames are pulled by the PC application by using a special “grab” command, as described in Section 6.5.

7.5 FPGA Frame FIFO

The frame FIFO is used to buffer up 1024 I/Q samples which are then sent up to the PC. Whenever a “grab” command occurs, as described in Section 6.5, the FPGA begins storing samples into the frame FIFO. Immediately after the first sample is stored, the UART interface begins sending sample bytes up to the PC. It is assumed that the UART will be slower at taking samples out of the FIFO than the ADC Interfaces is receiving them, such that an underflow condition is impossible. After all 1024 samples are sent up to the PC, the frame FIFO remains empty until the next “grab” command occurs.

8 FMC-1RX Command Protocol Specification

Note: The following section is an excerpt from the Bitshark FMC-1RX User's Manual, which can be found at [2]. It applies to the FMC-1RX card directly, as referenced from the FPGA Mezzanine Card (FMC) host. If interfacing with an FPGA platform hosting the FMC-1RX card, all commands must be preceded with the 'B' routing character (see Section 6.4 for details).

8.1 Overview

The Bitshark FMC-1RX utilizes a UART control interface to allow a user to configure the operation of the card. There are three electrical signals associated with this UART: transmit (HOST_UART_TX), receive (HOST_UART_RX), and ground (GND). All signals are referenced with respect to the FMC host (i.e., the HOST_UART_TX signal is a transmit signal from the FMC host to the FMC-1RX card). The high-level requirements of this interface are listed below:

- The UART interface operates at 9600 baud, with 8 data bits, 1 stop bit, no parity, and no flow control.
- Commands are sent and received from the FMC-1RX card as simple human-readable ASCII text.
- The ASCII text command interface is case-insensitive, as all commands are converted to uppercase by the FMC-1RX upon reception.
- The FMC-1RX card immediately echos all received characters back over its UART interface to confirm reception.
- All commands must be terminated with either a '\r' (carriage return) character or a '\n' (newline) character.
- Upon reception of either the '\r' or '\n' terminating character, the FMC-1RX card will echo back “\r\n”. This is the only case where the echo is not exactly what is received.
- Responses from the FMC-1RX consist of one or more lines that are terminated with a “\r\n” character sequence.
- All commands result in a response that contains, at minimum, an ACK or NCK indicating the command was acknowledged or not acknowledged.
- Information messages may be sent by the FMC-1RX as part of a response (in addition to the actual contents of the response). These messages begin with “I:”, contain non-critical details of the response, and terminate in “\r\n”.
- Error messages may be sent by the FMC-1RX as part of a response. These messages begin with “E:”, contain an error message and terminate in “\r\n”.
- All commands, regardless of success or failure, result in a '>' character (without a terminating “\r\n”) being sent by the FMC-1RX card as the final character in a response to indicate that it is ready for a new command.

8.2 Command Specification

8.2.1 Read

Name	read
Format	read <parameter>
Description	The “read” command is used to read the currently set value for a specific parameter on the FMC-1RX card.
Example Usage:	>read freq ACK 1972.5 MHz >

8.2.2 Write

Name	write
Format	write <parameter> <value> <optional value 1> <optional value 2> <...>
Description	The “write” command is used to update the currently set value for a specific parameter on the FMC-1RX card.
Example Usage:	>write freq 2400.020 ACK >

8.2.3 Help

Name	help
Format	help
Description	The “help” command is used to provide a description of all supported commands and parameters for the FMC-1RX, along with a brief set of examples.
Example Usage:	>help ACK <text describing supported commands and parameters, along with examples> >

8.3 Parameter Specification

8.3.1 Frequency

Name	freq
Format	read freq write freq <desired frequency in MHz>
Description	The freq parameter defines the currently tuned RF frequency in MHz. This parameter has a range from 300 MHz to 4000 MHz, with a resolution of 1 KHz. The default value of this parameter is 2000.0 MHz.
Example Usage:	>read freq ACK 1972.5 MHz > >write freq 2400.032 ACK >

8.3.2 RF Gain

Name	rfgain
Format	read rfgain write rfgain <desired gain in dB>
Description	The rfgain parameter provides control over the amount of gain used in the RF front end, measured in dB. This gain is in addition to fixed RF gain in the front end. This parameter has a range from 0 dB to 31.5 dB, in 0.5 dB steps. The default value of this parameter is 15 dB.
Example Usage:	>read rfgain ACK 20.5 dB > >write rfgain 31.5 ACK >

8.3.3 Baseband Gain

Name	bbgain
Format	read bbgain write bbgain <desired gain in dB>
Description	The bbgain parameter provides control over the amount of gain used in the baseband, measured

	<p>in dB. This gain is in addition to fixed baseband gain in the front end. The range of this parameter is dependent on the currently set chanbw parameter. The following bbgain values are available:</p> <ul style="list-style-type: none"> • 660 KHz <= chanbw < 8000 KHz: 0 dB, 6 dB • 8000 KHz <= chanbw < 16000 KHz: -6 dB, 0 dB, 6 dB • 16000 KHz <= chanbw < 32000 KHz: -12 dB, -6 dB, 0 dB, 6 dB • 32000 KHz <= chanbw <= 56000 KHz: -18 dB, -12 dB, -6 dB, 0 dB, 6 dB <p>The default value of this parameter is 6 dB.</p>
Example Usage:	<pre>>read bbgain ACK 0 dB > >write bbgain -12 E: gain not supported for current chanbw NCK > >write bbgain 6 ACK ></pre>

8.3.4 Channel Bandwidth

Name	chanbw
Format	read chanbw write chanbw <desired channel bandwidth in KHz>
Description	The chanbw parameter provides control of the corner frequency of the baseband analog low-pass filter. Typically, the RF signal of interest is downconverted and centered at 0 Hz (zero-IF). The baseband low-pass filter is used to attenuate additional signals outside of the channel bandwidth of interest. In the standard zero-IF usage, the chanbw parameter defines 2x the value of the low-pass filter corner frequency. This parameter supports the following range of values: 660 KHz, 1320 KHz, 2000 KHz to 56000 KHz in 1000 KHz steps. The default value of this parameter is 56000 KHz.
Example Usage:	<pre>>read chanbw ACK 4000 KHz > >write chanbw 8000 KHz ACK ></pre>

8.3.5 A/D Sample Clock

Name	adclk
Format	read adclk write adclk <desired sample rate in hertz>
Description	The adclk parameter provides control over the sample rate clock used to digitize samples in the A/D converter. This parameter has a range of 5000000 Hz to 105000000 Hz, with a resolution of 1 Hz. The default value of this parameter is 5000000 Hz.
Example Usage:	>read adclk ACK 7680000 Hz > >write adclk 104000000 ACK >

8.3.6 IQ Imbalance Calibration

Name	iqcal
Format	read iqcal <RF frequency in MHz> write iqcal <RF frequency in MHz> <16-bit 'C' parameter> <16-bit 'D' parameter>
Description	<p>The iqcal parameters provide calibration values to optimize the I/Q imbalance of the received signal, and thus improving the sideband suppression. These parameters are used in conjunction with the IQ imbalance correction block in the host FPGA to provide the desired balancing operation. The algorithm used for this imbalance correction can be found in [8].</p> <p>Each FMC-1RX card is calibrated at production time in 50 MHz increments to determine the the 'C' and 'D' parameters (see [8] for details on these parameters). The 'C' and 'D' parameters are then stored in EEPROM on the FMC-1RX card. A typical end-user application would read these parameters out at start-up, and would write these calibration parameters to the associated IQ imbalance correction block in the host FPGA as each RF frequency of interest is tuned. Additional details of how this correction block in the FPGA operates can be found in Section 7.2.</p> <p>Note: under normal operating conditions, it shouldn't be necessary to perform a write operation on the iqcal data. The only time this would be necessary is in the case where an end-user would want to perform an update to the calibration table.</p>
Example Usage:	>read iqcal 830.0 ACK 0x03F4 0x0001 > >write iqcal 1972.5 0x0002 0x0001

	ACK >
--	----------

8.3.7 Temperature Sensor

Name	temp
Format	read temp (write operation is not allowed)
Description	The temp parameters provides a current temperature reading of the FMC-1RX card in degrees Celsius.
Example Usage:	>read temp ACK 35 deg C >

8.3.8 User Programmable Clock

Name	userclk
Format	read userclk write userclk <desired clock rate in hertz>
Description	The userclk parameter provides control over a user programmable clock that is accessible on the FMC interface for use by the host FPGA. This clock is phase coherent with the reference clock used on the FMC-1RX. This parameter has a range of 5000000 Hz to 200000000 Hz, with a resolution of 1 Hz. The userclock can be disabled by setting the parameter to 0 Hz. The default value of this parameter is 0 Hz (disabled).
Example Usage:	>read userclk ACK 43600000 Hz > >write userclk 156000000 ACK > >write userclk 0 ACK >

8.3.9 Clock

Name	clock
Format	read clock write clock <ref clock source> <ref clock frequency in MHz>

Description	<p>The clock parameter is used to control the source and frequency of the reference clock used by the FMC-1RX. The <ref clock source> field allows for selection between the on-board 26 MHz TCVCXO and the external clock input. The following range of values are supported for <ref clock source>:</p> <ul style="list-style-type: none"> • 0: external clock input • 1: on-board 26 MHz TCVCXO (default) <p>The <ref clock frequency in MHz> field allows for definition of the reference clock frequency. By default, the FMC-1RX uses the on-board TCVCXO, which operates at 26 MHz. When the <ref clock source> field is set to 0, the <ref clock frequency in MHz> parameter can range from 10 MHz to 52 MHz.</p>
Example Usage:	<pre>>read clock ACK 1 26.0 MHz > >write clock 0 52.0 ACK ></pre>

8.3.10 Clock Warp Voltage

Name	clkwarp
Format	<pre>read clkwarp write clkwarp <warp voltage in DAC steps></pre>
Description	<p>The clkwarp parameter is used to control the value of the voltage used to warp the on-board TCVCXO reference clock. This provides fine-grained control of the reference clock frequency over a limited range. The warp voltage is generated by a 12-bit DAC which is capable of outputting voltages between 0V and 3.3V. The warp voltage for the TCVCXO can be between 0.5 V and 2.5V, which corresponds to DAC values of 683 and 3413. Note: modifications to the warp voltage will effect all clocks derived from the reference clock, which include the RF center frequency, the A/D sample clock, and the user programmable clock. The resultant modification in frequency to the derived clock will depend on the current value of the derived clock.</p>
Example Usage:	<pre>>read clkwarp ACK 2430 > >write clkwarp 35 E: warp value out of range NCK ></pre>

```
>write clkwarp 988  
ACK  
>
```