Hardware Manual

SC5319A & SC5320A

20 GHz to 40 GHz RF Downconverter

Preliminary Rev 1.3

www.signalcore.com

Table of Contents

1 General Information

1.1 Warranty

This product is warranted against defects in materials and workmanship for a period of three years from the date of shipment. SignalCore will, at its option, repair or replace equipment that proves to be defective during the warranty period. This warranty includes parts and labor.

Before any equipment will be accepted for warranty repair or replacement, a Return Material Authorization (RMA) number must be obtained from a SignalCore customer service representative and clearly marked on the outside of the return package. SignalCore will pay all shipping costs relating to warranty repair or replacement.

SignalCore strives to make the information in this document as accurate as possible. The document has been carefully reviewed for technical and typographic accuracy. If technical or typographical errors exist, SignalCore reserves the right to make changes to subsequent editions of this document without prior notice to possessors of this edition. Please contact SignalCore if errors are suspected. In no event shall SignalCore be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, SIGNALCORE, INCORPORATED MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF SIGNALCORE, INCORPORATED SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. SIGNALCORE, INCORPORATED WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of SignalCore, Incorporated will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against SignalCore, Incorporated must be brought within one year after the cause of action accrues. SignalCore, Incorporated shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow SignalCore, Incorporated's installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

1.2 Copyright & Trademarks

Under the copyright laws, this publication may not be reproduced or transmitted in any form, electronic or mechanical, including photocopying, recording, storing in an information retrieval system, or translating, in whole or in part, without the prior written consent of SignalCore, Incorporated.

SignalCore, Incorporated respects the intellectual property rights of others, and we ask those who use our products to do the same. Copyright and other intellectual property laws protect our products. Use of SignalCore products is restricted to applications that do not infringe on the intellectual property rights of others.

“SignalCore”, “signalcore.com”, and the phrase “preserving signal integrity” are registered trademarks of SignalCore, Incorporated. Other product and company names mentioned herein are trademarks or trade names of their respective companies.

1.3 Warnings Regarding Use of SignalCore Products

(1) PRODUCTS FOR SALE BY SIGNALCORE, INCORPORATED ARE NOT DESIGNED WITH COMPONENTS NOR TESTED FOR A LEVEL OF RELIABILITY SUITABLE FOR USE IN OR IN CONNECTION WITH SURGICAL IMPLANTS OR AS CRITICAL COMPONENTS IN ANY LIFE SUPPORT SYSTEMS WHOSE FAILURE TO PERFORM CAN REASONABLY BE EXPECTED TO CAUSE SIGNIFICANT INJURY TO A HUMAN.

(2) IN ANY APPLICATION, INCLUDING THE ABOVE, RELIABILITY OF OPERATION OF THE SOFTWARE PRODUCTS CAN BE IMPAIRED BY ADVERSE FACTORS, INCLUDING BUT NOT LIMITED TO FLUCTUATIONS IN ELECTRICAL POWER SUPPLY, COMPUTER HARDWARE MALFUNCTIONS, COMPUTER OPERATING SYSTEM SOFTWARE FITNESS, FITNESS OF COMPILERS AND DEVELOPMENT SOFTWARE USED TO DEVELOP AN APPLICATION, INSTALLATION ERRORS, SOFTWARE AND HARDWARE COMPATIBILITY PROBLEMS, MALFUNCTIONS OR FAILURES OF ELECTRONIC MONITORING OR CONTROL DEVICES, TRANSIENT FAILURES OF ELECTRONIC SYSTEMS (HARDWARE AND/OR SOFTWARE), UNANTICIPATED USES OR MISUSES, OR ERRORS ON THE PART OF THE USER OR APPLICATIONS DESIGNER (ADVERSE FACTORS SUCH AS THESE ARE HEREAFTER COLLECTIVELY TERMED “SYSTEM FAILURES”). ANY APPLICATION WHERE A SYSTEM FAILURE WOULD CREATE A RISK OF HARM TO PROPERTY OR PERSONS (INCLUDING THE RISK OF BODILY INJURY AND DEATH) SHOULD NOT BE SOLELY RELIANT UPON ANY ONE COMPONENT DUE TO THE RISK OF SYSTEM FAILURE. TO AVOID DAMAGE, INJURY, OR DEATH, THE USER OR APPLICATION DESIGNER MUST TAKE REASONABLY PRUDENT STEPS TO PROTECT AGAINST SYSTEM FAILURES, INCLUDING BUT NOT LIMITED TO BACK-UP OR SHUT DOWN MECHANISMS. BECAUSE EACH END-USER SYSTEM IS CUSTOMIZED AND DIFFERS FROM SIGNALCORE'S TESTING PLATFORMS, AND BECAUSE A USER OR APPLICATION DESIGNER MAY USE SIGNALCORE PRODUCTS IN COMBINATION WITH OTHER PRODUCTS IN A MANNER NOT EVALUATED OR CONTEMPLATED BY SIGNALCORE, THE USER OR APPLICATION DESIGNER IS ULTIMATELY RESPONSIBLE FOR VERIFYING AND VALIDATING THE SUITABILITY OF SIGNALCORE PRODUCTS WHENEVER SIGNALCORE PRODUCTS ARE INCORPORATED IN A SYSTEM OR APPLICATION, INCLUDING, WITHOUT LIMITATION, THE APPROPRIATE DESIGN, PROCESS AND SAFETY LEVEL OF SUCH SYSTEM OR APPLICATION.

2 Physical Description

2.1 Unpacking

All SignalCore products ship in antistatic packaging (bags) to prevent damage from electrostatic discharge (ESD). Under certain conditions, an ESD event can instantly and permanently damage several of the components found in SignalCore products. Therefore, to avoid damage when handling any SignalCore hardware, you must take the following precautions:

1. Ground yourself using a grounding strap or by touching a grounded metal object.
2. Touch the antistatic bag to a grounded metal object before removing the hardware from its packaging.
3. **NEVER touch exposed signal pins.** Due to the inherent performance degradation caused by ESD protection circuits in the RF path, the device has minimal ESD protection against direct injection of ESD into the RF signal pins.
4. When not in use, store all SignalCore products in their original antistatic bags.

Remove the product from its packaging and inspect it for loose components or any signs of damage. Notify SignalCore immediately if the product appears damaged in any way.

2.2 Setting Up the Device

Integration of the SC5319A and SC5320A modules requires attention to maintain effective cooling. Inadequate cooling can cause the temperature inside the RF housing to rise above the maximum for this product, leading to improper performance, reduction of product lifespan, or complete product failure. SignalCore suggests providing moderate airflow across the RF housing. If active cooling is not an option, use thermal interface materials to bond the RF housing to a larger heatsinking surface (i.e. a system enclosure). As each device’s integrated system configuration is unique, detailed cooling options cannot be provided.

A cooling plan is sufficient when the SC5319A and SC5320A on-board temperature sensors indicate a rise of no more than 20°C above ambient temperature under normal operating conditions.

2.3 Front Interface Indicators and Connectors

The SC5319A is a single slot PXIe formfactor module, whose front face in shown below in Figure 1.

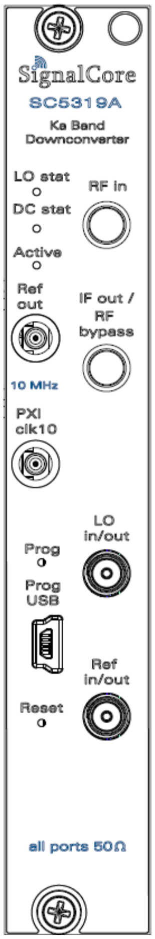

Figure 1 The SC5319A is a PXIe-based RF downconverter with all user I/O located on the front face of the module.

The SC5320A is a serial controlled core module with all user connections on the front face of the module.

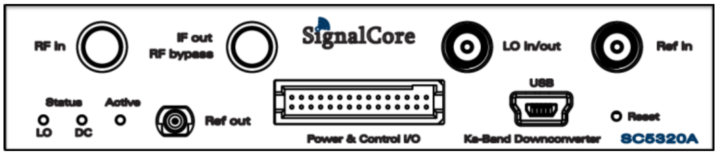

Figure 2. The SC5320A USB and serial interface module front face

2.3.1 Signal Connections

The RF in and IF out connectors are 2.92mm K type, and the LO in/out and Ref in are SMA type. The Ref out connector is an MMCX type. Exercise caution when fastening cables to the signal connections, it is important not to twist and turn the cable too tight on the K or SMA connectors, rather the cables should be held still while the connector nut is turned. Twisting and turning of the cable could put sufficient force on the center pins of both the male and female ends, causing them to tear away from their soldered joints. Furthermore, over-tightening any connection can also apply torque on the center pins and threads, causing permanent damage to the device.

⚠ *The condition of your system signal connections can significantly affect measurement accuracy and repeatability. Connections that are improperly mated, dirty, damaged, or worn can degrade measurement performance. Clean out any loose, dry debris from connectors with clean, low-pressure air (available in spray cans from office supply stores).*

⚠ *If deeper cleaning is necessary, use lint-free swabs and isopropyl alcohol to gently clean inside the connector barrel and the external threads. Do not mate connectors until the alcohol has completely evaporated. Excess liquid alcohol trapped inside the connector may degrade measurement performance until fully evaporated (this may take several days).*

⚠ *Tighten all SMA connections with 3 in-lb min to 5 in-lb max (56 N-cm max)*

RF In	This is the RF input port to the device with nominal impedance of 50 Ω. Its maximum input power is +27 dBm.
IF out / RF bypass	This port is the IF output under normal conversion operation. However, the RF conversion can be bypassed and switch directly from the RF input port to here.
LO in/out	When the conversion LO is selected to use the internal synthesizer, the synthesized signal is export out via this port, otherwise this port takes an external source as the LO.
Ref In (Out)	This is the 10 MHz reference input to the device, enabling the device to phase lock its internal clocks to an external reference source. The port may be programmed to export out the device 10 MHz reference clock.
Ref Out	This is a dedicated 10 MHz reference output, a duplicate signal of the device on board reference clock.
PXI clk10	This is the exported 10 MHz of the reference for the PXIe chassis.

2.3.2 Device LED Indicators

These are LED indicator lights for the device, and their functions are listed in *Table 1* and *Table 2*.

Table 1. Status LED Indicators

LED Color	Description
Green	The device circuitry is functioning properly in the state that it is programmed for.
Amber	Indicates that all functions are on standby mode.
Red	Indicates that one or more local oscillators are not functioning correctly, or detected fault occurred.
Off	No supply or supply error

Table 2. Active Indicator

LED Color	Description
Green	An external interface port has accessed the device via API.
Red	Input supply voltage or device current exceeded. The device is in full power down state.
Off	No current interface access via API.

2.3.3 Communication and Supply Connection

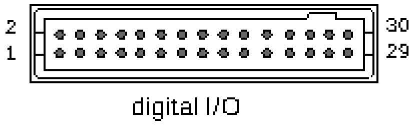

Figure 3. Power and Digital IO Connector

Power and communication to the SC5320A is provided through a rectangular connector from Samtec whose part number is TFM-115-01-L-D-RA. It also serves as the digital connector interface for RS232/SPI, trigger, and other digital signals. With an exception for the RS232 logic levels, all other logic levels are 3.3V LVCMOS whose inputs are also 5V tolerant. A mating pig-tail cable, part number SFSD-15-28-H-10.00-SR, is provided with the product. The pin definitions are listed in *Table 3*.

⚠ *Pinouts are different for different SignalCore products with the same connector type. Please ensure that mating connectors and cables are wired correctly before connection.*

Table 3. Interface connector pin out description

PIN #	Description
2,4,6,8	12V Supply Rail
12	Converter Status/Active
14	System Reset, Logic 0 to Reset Device
16	RS232 Baud Rate: 1(default) -> 115200, 0 -> 57600 SPI Mode Select: 1 (default) mode 1, 0 mode 0
17	Reserved, Pull High to 3.3V or DNC
19	Device Accessed/Active
20	LO Status/Active
24	TX for RS232 or MOSI for SPI (Host referred)
26	SPI CS_b
27	RX for RS232 or MISO for SPI (Host referred)
30	SPI CLK
9,10,13, 18,21,22	Reserved, Do Not Connect
1,3,7,11,15,23,29	GND

2.3.4 Mini-USB Connection

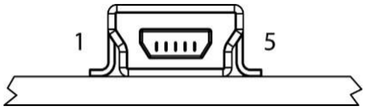

Figure 4

This is a mini-USB Type B connector for USB communication with the device using the standard USB 2.0 protocol (full speed) found on most host computers. This connector is also used for updating firmware in the field and is exclusively used for this purpose on the SC5319A. The pinout of this connector, viewed from the front, is shown in the following table.

PIN #	USB Function	Description
1	VBUS	Vcc
2	D-	Serial Data (neg)
3	D+	Serial Data (pos)
4	ID	Not Used
5	GND	Device Ground (also tied to connector shell)

2.3.5 Reset Button (Pin Hole)

Behind this pin hole is the reset button, which is only available on the SC5320A. Using a pin to lightly depress this momentary-action push button switch will cause a hard reset to the device, putting it back to its default settings. All user settings will be lost. System reset capability can also be accessed through the communication header connector.

2.3.6 The Prog Button (Pin Hole)

Behind this pin hole is the button to put the device into firmware programming mode. On the SC5320A, this pin hole is located on the right side of the device when viewing from the front. On the SC5319A the pin hole is labelled “Prog”. Exercise care when accessing this button because the button is depressed accidentally, it could wipe out the device firmware rendering it inoperable until firmware is restored.

3 Functional Description

3.1 Overview

The SC5320A uses USB as its primary interface with an optional SPI or RS232 interface. The SC5319A is a PXIe version of the product.

The downconverter assembly consists of 2 module parts:

- The Signal Conversion Module contains the mixers, filters, signal amplifiers, and attenuators. This module is referred to as the "signal chain". This conversion module also includes a built-in LO synthesizer.

- The Power Conditioning and Digital Control Board contains the supply switchers that generate the needs rails for the RF modules and an onboard MCU that provides both the computation engine and interface between the user and the RF modules.

The figure below shows how the modules relate to each other.

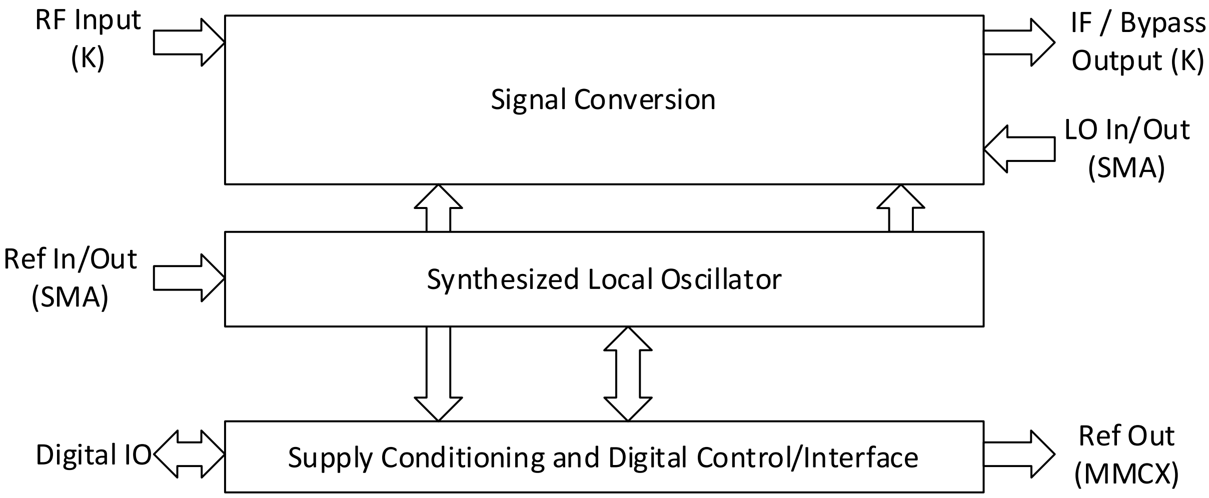

Figure 5. Simplified Block Representation of the Downconverter Assembly Module

The downconverter has a single conversion stage that converts the RF signal to an intermediate frequency (IF). The RF input range of this downconverter is from 20 GHz to 40 GHz, while the output IF range is from 50 MHz to 5000 MHz. The LO signal needed for the conversion may be selected from the internal synthesizer or an external signal source. The range of the LO is from 10 GHz to 20 GHz. The frequency accuracy and stability of the internal LO is derived from an onboard Oven Controlled Crystal Oscillator (OCXO) with an initial accuracy better than 20 ppb. For better accuracy and stability, the synthesizer can lock to an external reference of higher precision via the Ref In port.

3.2 The Signal Chain

The conversion module contains the mixer, filters, amplifiers, and attenuators used to convert, purify, and maintain the conversion gain of the signal between the input of the device to its output.

This cascade of signal conversion and conditioning stages is often referred to as the signal chain, which is shown on the system block drawing of Figure 6.

3.2.1 The RF Input

The RF port connector is a 2.92mm (K) type rated to 40 GHz of operation. The RF port is AC coupled with a 0.1 uF broadband capacitor as indicated in Figure 6. The input frequency range at this port is from ~100 kHz to 40 GHz; the full range can be switched directly to the IF port, however 20 GHz to 40 GHz signals can be converted (see 3.2.3). The usability of the upper out-of-bounds region depends on the roll-off response of the mixer.

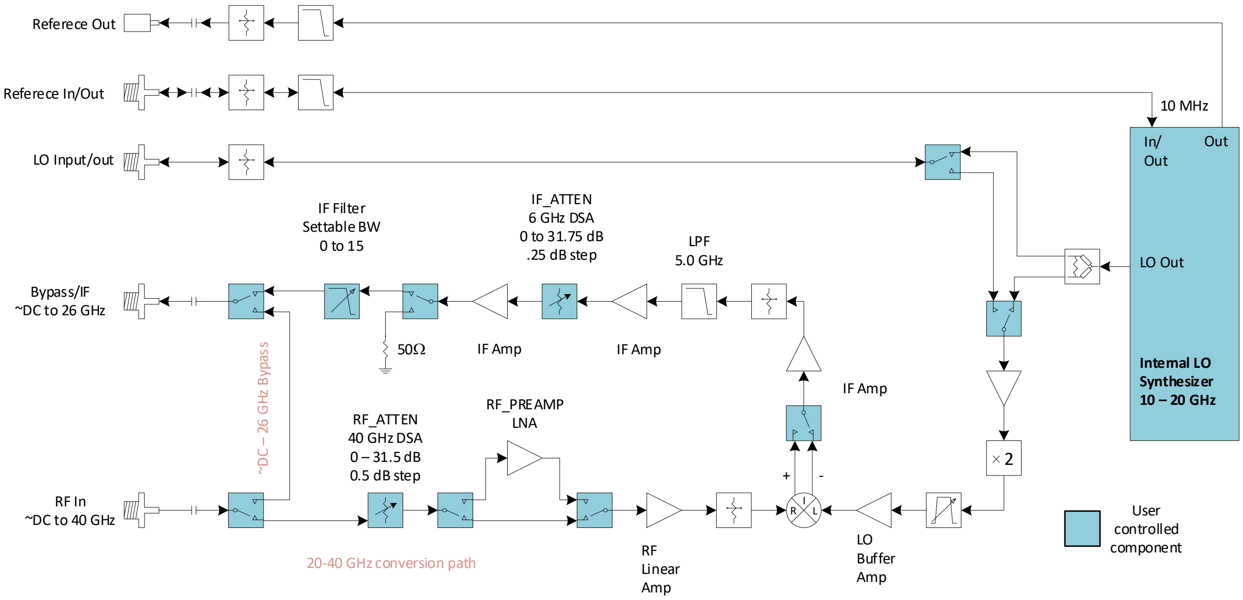

Figure 6. Block diagram of the downconverter assembly

3.2.2 The LO Input/Output

The LO input/out port connector is an SMA type. When the device is configured to use its internal synthesizer as the LO for the mixer, this port is enabled as an output for the synthesized signal. In other words, the device can export its LO signal out to be used by other devices. This is useful in phase coherent multiple channel systems that uses a common LO. This port can also be configured as an input for an external LO source; in this case the internal synthesizer is disabled and powered down to standby.

3.2.3 The Input Path Switch

Immediately following the capacitor is an RF switch to selectively direct the RF signal down two paths:

1. The default is the conversion path where the RF signal is converted to an IF.
2. The alternate path is directly to the IF output port, bypassing all means of frequency conversion. The upper frequency limit of this bypass path is 26 GHz.

When the conversion path is bypassed, the IF output is disabled and the internal LO is powered down.

3.2.4 The RF Conversion Path

The first device in the RF conversion path is a step attenuator with 0.5 dB step resolution. This attenuator (RF_ATTEN) is used to control the signal level at the mixer or at the RF preamplifier if it is enabled. If the RF level at the input of these devices is relatively large with respect to their input compression points, the signal would experience significant nonlinear effects that would distort its waveform and produce large spurious 3rd intermodulation distortion effects.

Following RF_ATTEN is a switchable RF amplifier that can be switched into the signal path to improve the device sensitivity, effectively lowering the device noise figure. In other words, the effective input noise level of the device is lowered when this amplifier is enabled. When signals with levels lower than -30 dBm are expected, turning on this amplifier is recommended to improve the device sensitivity to these levels. The typical maximum gain of the downconverter without the amplifier enabled is +30 dB, and when the amplifier is enabled, its additional +20 dB will boost the typical maximum gain to approximately +50 dB.

The RF linear amplifier prior to the mixer input acts as a unity buffer to improve the reverse transmission of signals at the mixer or created at the mixer such as the LO and IF products. Another point to mention is that there are no band selective filters in the RF path, so in open environments the device is unable to differentiate an RF signal from its image. A band select filter at the RF input port is recommended to properly filter through the signal of interest before down conversion.

3.2.5 The First Mixer and IF Path

The mixer (RF mixer) of the downconverter is extremely critical as it sets the dynamic ranges of the device, for both the signal-to-noise (DR_{SNR}) and third order IMD (DR_{IMD}) dynamic ranges. The DR_{SNR} and DR_{IMD} are directly related to the mixer input compression point (IP1dB) and input third-order intercept point (IIP3). The IIP3 of the mixer is typically better than +12 dBm, and its input P1dB is better than 5 dBm. In applications that require better signal linearity, it is recommended that the level at the mixer input is kept at -20 dBm or less. Typically, the IMD is better than -65 dBc for two -20 dBm tones at the mixer (no RF attenuation). Increasing the level at the mixer improves SNR of the IF output signal at the expense of higher nonlinear 3rd order byproducts.

Another important characteristic of the mixer is the LO-IF isolation and LO-RF isolation. The higher the isolation, the lower the leakage LO signal is at the RF and IF ports. The RF mixer is driven with a ~+17 dBm LO signal and the LO-IF isolation is about 45 dB, resulting in a -28 dBm LO leakage at the IF terminal of the mixer, which may leak into the IF path. The low-pass IF filters suppress this leakage to levels well below -60 dBm. The LO-RF isolation is also about 45 dB so the LO leakage at the RF port of the mixer is on the order of -28 dBm. The high reverse isolation of the RF linear power amplifier suppresses this LO leakage to < 60 dBm. When the low noise pre-amplifier is enabled, it will provide another 30 dB of reverse isolation, lowering the LO leakage at the RF input port to < 90 dBm.

The frequency relationship between the three ports of the mixer is given as:

$$IF = \begin{cases} LO - RF \\ RF - LO \end{cases}$$

From the above equation, the IF output spectrum is inverted with respect to the RF in the first relationship, whereas in the second it is noninverted. The former is lower sideband conversion, and the latter is upper sideband conversion. When the LO > RF, the IF signal is inverted, and it is not inverted when LO < RF.

3.2.6 Signal Chain Configurations

Recommended signal chain configurations are provided below for various application scenarios. These are strictly recommendations. They are not indicative of the downconverter function limitations. Users will need to adapt the device to their applications at hand.

3.2.6.1 *Signals Bypassing the Conversion Stage*

The RF input signal can be directed to the RF output port, bypassing the conversion process, and hand the signal to another downconverter that might be able to cover bands outside of the device range. For example, the SC5318A downconverter mates well with this device to form an integrated converter with continuous frequency coverage from 6 GHz to 40 GHz.

3.2.6.2 *Dynamic Range Setting*

There are 2 digital step attenuators to control the conversion gain of the downconverter. The first set consists of RF_ATTEN in the RF stage and the second set consists of IF_ATTEN in the IF stage. Both attenuators have 30 dB of settable range, however the IF attenuator has fine resolution of 0.25 dB while the RF attenuator has resolution of 0.5 dB.

To set the downconverter for better sensitivity or better SNR, the gain should be shifted to the RF input path of the device before the mixer. The RF pre-amplifier should be enabled if necessary and/or RF attenuation reduced. The IF attenuator is then used to adjust the final IF output level. The drawback is that the signal level starts off higher as it enters the first mixer as well as the subsequent components so, as a result, the apparent linearity of the device is lower.

To set the device for better linearity, the gain should be shifted to the output IF path (after the mixer) and reduced in the RF path. The signal power level at the input should be lower than -20 dB for improved linearity. Since the input signal is low, the relative SNR will be lowered too. But, as the first mixer and subsequent components experience lower power levels, the apparent linearity of the device is improved at the expense of a lower SNR dynamic range.

When the device gain is balanced well, the device could achieve SNR better than 130 dBc/Hz while maintaining IMD3 levels close to 68 dBc. These numbers are representative of converters used in large box high end spectrum analyzers. When the device is optimized for best SNR, typical values greater than -150 dBc/Hz can be achieved, and when the device is optimized for sensitivity by enabling the RF preamplifier, the input spectral noise floor is typically lower than -165 dBm/Hz. The flexible use of these attenuators and pre-amplifier allows the downconverter to achieve better than 190 dB of measurement dynamic range.

3.3 The LO Synthesizer

The internal LO synthesizer is a hybrid between integer-N PLL and DDS, enabling it to tune at 1 Hz steps while maintaining low phase noise. The master reference signal for the generation of the LO signal comes either from an internal Oven Controlled Crystal Oscillator (OCXO) or an external signal source as shown below in Figure 7. The colored blocks indicate that they are user controllable via software. The frequency range of the synthesizer is 10 to 20 GHz, which is frequency doubled to drive the mixer LO port.

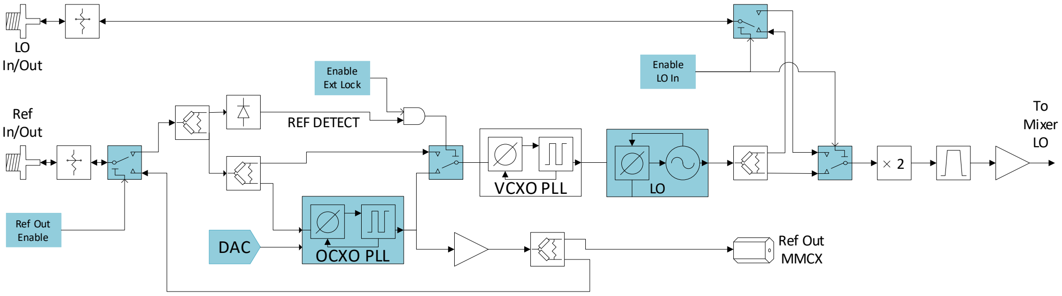

Figure 7. Block diagram of the local oscillator

3.3.1 The Reference Clocks

The internal base clock of the downconverter is a 10 MHz Voltage Controlled Oven Controlled Crystal Oscillator (VCOCXO) with initial accuracy better than 20 ppb once the device has reached a stable temperature. Its initial accuracy is set at the factory via an on-board 14-bit voltage reference DAC. This DAC is accessible for dynamic accuracy calibration. The other reference is a 100 MHz Voltage Controlled Crystal Oscillator (VCXO), which is phase locked to this base reference whenever an external reference source is not used.

When an external reference is selected as the base clock, by enabling the device to phase lock to it, the device will only attempt to lock when the presence of a reference signal is detected at the input port. This sets the device to “phase lock on detection”. Notice that both the reference clocks (OCXO and VCXO) will attempt to lock to the external source to ensure there always is only one base frequency source. If an external source is not used, its connection should physically be removed from the input port to avoid the device from picking on its signal as this could produce unwanted offset spurious effects. Having the VCXO lock directly to the external source has the advantage of utilizing the close-in phase noise of the source; this is assuming the external source phase noise is superior to the internal base. Finally, although the internal VCOCXO is not used when an external reference is selected, it remains powered on and locked to maintain its frequency coherence and temperature stability.

3.3.2 The Reference Ports

There are 2 reference ports on this device. First, there is a SMA type connector whose main purpose is to receive an external reference signal. However, it can be programmed as an output of the internal OCXO reference signal. Second, the other is a MMCX type connector that is always active as an output.

3.3.3 LO Port

The LO port is bi-directional; it may be programmed either as an input or output. The frequency range is 10 GHz to 20 GHz.

3.4 The Interface Module

The following figure shows the power and control interface block for the device. All supply rails are produced on board; their voltages are regulated and actively filtered to keep noise to a minimum. Thus, these downconverters are tolerant to “dirty” external power supplies.

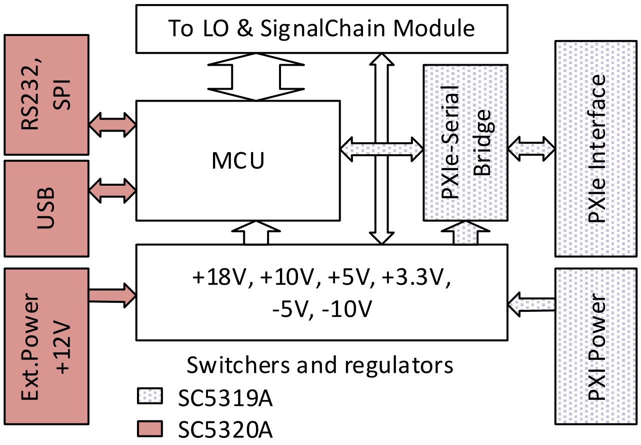

Figure 8. Block diagram of the power and interface module

4 Hardware Registers

The set of hardware registers of the downconverter may be divided into a configuration set and a query set; the configuration registers are write-only registers to set up the states of the device, while the query registers request the device to prepare and send back data associated with them. The registers are identical for all communication interfaces: PXIe, USB, RS232, and SPI. Data communication to all interfaces are sent in 1 byte wide, that is, data is transferred byte-by-byte. Register data lengths vary in size, and it is important that the correct number of bytes are sent to the corresponding register. Failure to do so may cause communication to lock-up and the device to become unresponsive.

4.1 Configuration Registers

These are write-only registers to configure the device. The registers vary in length to reduce redundant data and improve the communication speed, especially for SPI and RS232 interfaces. Furthermore, it is vitally important that the length of data written to a register is exact because failure to do so will cause the interfaces to misinterpret the incoming data, leaving the device in a stalled state. The total number of bytes is the sum of the register address (1 byte) and its corresponding data bytes. For example, to set the RF frequency value, eight bytes must be written; the sum of the 1 register byte and 7 data bytes. See the RF_FREQUENCY register of Table 4. The table provides a summary of the configuration registers, and each register is explained in detail.

Table 4. Configuration Registers

Register Name	Reg Add	Serial Range	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INITIALIZE	0x01	[7:0]	Set to zeros [7:1]				Mode			
SYSTEM_ACTIVE	0x02	[7:0]	Set to zeros [7:1]				Enable “active” LED			
SYNTH_MODE	0x03	[7:0]	Set to zeros [7:2]			Fast-tune		Loop Gain		
RF_FREQUENCY	0x10	[7:0]	Frequency Word (Hz) [7:0]							
		[15:8]	Frequency Word (Hz) [15:8]							
		[23:16]	Frequency Word (Hz) [23:16]							
		[31:24]	Frequency Word (Hz) [31:24]							
		[39:32]	Frequency Word (Hz) [39:32]							
		[47:40]	Frequency Word (Hz) [47:40]							
		[55:48]	Frequency Word (Hz) [55:48]							
IF_FREQUENCY	0x11	[7:0]	Frequency Word (Hz) [7:0]							
		[15:8]	Frequency Word (Hz) [15:8]							
		[23:16]	Frequency Word (Hz) [23:16]							
		[31:24]	Frequency Word (Hz) [31:24]							
		[39:32]	Frequency Word (Hz) [39:32]							
		[47:40]	Frequency Word (Hz) [47:40]							
		[55:48]	Frequency Word (Hz) [55:48]							
LO_FREQUENCY	0x12	[7:0]	Frequency Word (Hz) [7:0]							
		[15:8]	Frequency Word (Hz) [15:8]							
		[23:16]	Frequency Word (Hz) [23:16]							
		[31:24]	Frequency Word (Hz) [31:24]							
		[39:32]	Frequency Word (Hz) [39:32]							
		[47:40]	Frequency Word (Hz) [47:40]							
		[55:48]	Frequency Word (Hz) [55:48]							
LO_SOURCE	0x14	[7:0]	Set to zeros [7:1]							Ext
RF_AMP	0x14	[7:0]	Set to zeros [7:1]							Enable
ATTENUATOR	0x15	[7:0]	Attenuator Value							

Register Name	Reg Add	Serial Range	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		[15:8]	Set to zeros [15:11]				Attenuator Number			
		[23:16]	Set to zeros [23:16]							
BY_CONVERSION	0x16	[7:0]	Set to zeros [7:1]							Bypass convert
IF_OUT_ENBLE	0x16	[7:0]	Set to zeros [7:1]							Enable
IF_FILTER_SELECT	0x18	[7:0]	Set to zeros [7:4]				Filter number 0 - 15			
IF_SB_SELECT	0x19	[7:0]	Set to zeros [7:1]							Upper
REFERENCE_MODE	0x1A	[7:0]	Set to zeros [7:3]					Ref Dir	PXI 10 enable	Lock ext
REFERENCE_ADJUST	0x1B	[7:0]	DAC WORD [7:0]							
		[15:8]	zeros [15:14]		DAC WORD [13:8]					
		[23:16]	Set to zeros							
STORE_DEFAULT_STATE	0x1C	[7:0]	Set to zeros [7:0]							
SYNTH_SELF_CAL	0x1D	[7:0]	Set to zeros [7:1]							VCO#
DEVICE_STANDBY	0x1E	[7:0]	Set all to zeros						Standby Mode	
AUTO_CONV_PARAMS	0x1F	[7:0]	Linear mode			Linear mode			Auto amp ctrl	HW auto conv
		[15:8]	IF level (dB) [7:0]							
		[23:16]	Sign	IF level (dB) [14:8]						
		[31:24]	Mixer level (dB) [7:0]							
		[39:32]	Sign	Mixer level (dB) [14:8]						
		[47:40]	RF level (dB) [7:0]							
		[55:48]	Sign	RF level (dB) [14:8]						

4.1.1 Register 0x01 INITIALIZE

This register initializes the devices to the default state or reprograms all the device components with the current state parameters. Note that this register does not need to be called to program the device.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[0]	WO	Mode	1	0 = Resets the current state 1 = Resets the default to the default or startup state
[7:1]	WO	Unused	7	Set to zeros
[7:0]	RO	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.2 Register 0x02 SYSTEM_ACTIVE

This register turns on or off the active LED indicator on the front connector interface of the device. This register should be called when the device is opened or closed in software.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[0]	WO	Mode	1	0 = Turns off the active LED 1 = Turns on the active LED
[7:1]	WO	Unused	7	Set to zeros
[7:0]	RO	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.3 Register 0x03 SYNTH_MODE (2 Bytes)

This register configures the PLL loop gain of the local oscillator synthesizers. It also enables or disables faster tuning of the VCO based oscillator of LO1.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[1:0]	WO	Loop Gain	2	0 = Low loop gain, improves phase noise > 50 kHz 1 = Normal loop gain 2 = High loop gain, improves phase noise < 50 kHz
[2]	WO	Fast Tune	1	0 = Turns fast tune off, default 1 = Turns fast tune on, close in phase noise may degrade
[7:3]	WO	Unused	5	Set to zeros
[7:0]	RO	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.4 Register 0x10 RF_FREQUENCY

This register tunes the device to the input RF frequency.

Bytes written	8
Bytes read	1

Bits	Type	Name	Width	Description
[55:0]	W	Frequency word	56	Frequency word in Hz.

Bits	Type	Name	Width	Description
[7:0]	RO	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.5 Register 0x11 IF_FREQUENCY

This register sets the final IF value.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[55:0]	W	Frequency word	56	Frequency word in Hz.
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.6 Register 0x12 LO_FREQUENCY

This register sets the final IF value.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[55:0]	W	Frequency word	56	Frequency word in Hz.
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.7 Register 0x13 LO_SOURCE

This register selects the LO source from either the internal synthesizer or an external source.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[0]	W	LO source select	1	0 = Use internal synthesizer 1 = Use external source
[7:1]	W	Unused	7	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.8 Register 0x14 RF_AMP

This register enables and disables the RF amplifier.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[0]	W	RF Preamp Enable	1	0 = Disables the RF amplifier 1 = Enables the RF amplifier
[7:1]	W	Unused	7	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.9 Register 0x15 ATTENUATOR

This register sets the value of the device attenuators.

Bytes written	4
Bytes read	1

Bits	Type	Name	Width	Description
[7:0]	W	Attenuator value	8	In 0.25 dB LSB for IF, 0.5 dB LSB for RF
[10:8]	W	The target attenuator	3	The attenuator number: 0 = RF Atten 1 = RF Atten
[23:11]	W	Unused	13	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.10 Register 0x16 BYPASS_CONVERSION

This register configures the device to bypass the conversion path and direct the RF input port to the IF output port.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[0]	W	Bypass Converter	1	0 = Normal conversion path 1 = Bypasses the conversion, switches RF input directly to RF output
[7:1]	W	Not used	7	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.11 Register 0x17 IF_OUT_ENABLE

This register enables the IF output.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[0]	W	IF enable	1	0 = IF output is disabled, signal is shunt to ground. 1 = If Output is enabled
[7:1]	W	Not used	7	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.12 Register 0x18 IF_FILTER_SELECT

There are 16 (0 to 15) programmable low-pass bandwidths for the IF filter.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[7:0]	W	Filter number	8	Number of the filter bandwidth. The largest bandwidth is 15 and 0 is the narrowest.
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.13 Register 0x19 IF_SIDEBAND_SEL

This register selects either the upper sideband conversion or the lower sideband conversion.

Bytes written	2

Bytes read	1

Bits	Type	Name	Width	Description
[0]	W	Sideband	1	0 = Lower sideband conversion 1 = Upper sideband conversion
[7:1]	W	Unused	4	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.14 Register 0x1A REFERENCE_CLOCK

This register configures the reference clock behavior.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[0]	W	LockEnable	1	0 = Uses internal 10 MHz TCXO
[1]	W	PXI10Enable	1	Only on SC5319A to enable export of the 10 MHz backplane clock, this bit is ignored in the SC5320A.
[2]	W	REF Direction	1	Set the reference port (SMA) to as input or output 0 = input 1 = output
[7:3]	W	Unused	5	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.15 Register 0x1B REFERENCE_ADJUST

This register adjusts the 10 MHz OCXO accuracy by writing its control DAC value.

Bytes written	4
Bytes read	1

Bits	Type	Name	Width	Description
[13:0]	W	Tuning DAC word	14	DAC WORD
[23:14]	W	Unused	10	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.16 Register 0x1C STORE_DEFAULT_STATE

This register will store all current settings of the device into EEPROM and will become the default startup setting.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[7:0]	W	Unused	8	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.17 Register 0x1D SELF_SYNTH_CAL

This register forces an internal calibration of the synthesizer oscillators

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[0]	W	Oscillator Select	1	0 = Coarse VCO, 1 = Sum VCO
[7:1]	W	Used	7	EEPROM address
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.18 Register 0x1E DEVICE_STANDBY

This register sets either the entire device or sections of the device into standby mode. Placing a section into standby involves powering down its circuitry. This conserves power and eliminates unwanted LO signals when they are not in use.

Bytes written	2
Bytes read	1

Bits	Type	Name	Width	Description
[1:0]	W	Mode	2	0 = Device is fully power 1 = Powers down the synthesizer only 2 = Powers down the device
[7:2]	W	Unused	4	Set to zeros
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.1.19 Register 0x1F AUTO_CONV_PARAMS

The device uses these desired parameters to automatically determine the path, attenuator, and preamplifier settings that will best meet their requirements.

Bytes written	8
Bytes read	1

Bits	Type	Name	Width	Description
[0]	W	AUTO Conv Enable	1	Software use, bit to enable computation of gain and setting of it.
[1]	W	Auto Preamp Ctrl	1	0 = Preamplifier states is unchanged by the algorithm. 1 = Automatically changes the state of the pre-amplifier to meet the requirements.
[4:2]	W	Linearity State	3	0 = use mixer level 1 = balance SNR and linearity 2 = better SNR 3 = best SNR 4 = better linearity 5 = best linearity
[5]	W	HW Auto Enable	1	Enables computation in device, may slow the device from changing frequencies due to heavy computation. This bit is normally set to 0 if the API is used or computation occurs on the host computer to configure the device. If this bit is set to 1, the device will perform the computation to configure the device automatically only when register 0x30 is called with parameter value 13. If this bit is set to 0 and register 0x30 is called, the gain is computed based on the last configuration.
[7:6]	W	unused	2	Set to zeros
[14:8]	W	Absolute IF level	15	IF level in dB steps
[15]	W	Sign of IF level	1	0 = positive 1 = negative
[22:16]	W	Absolute mixer level	15	Mixer level in dB steps
[23]	W	Sign of mixer level	1	0 = positive 1 = negative
[30:24]	W	Absolute RF level	15	Mixer level in dB steps
[31]	W	Sign of RF level	1	0 = positive 1 = negative
[7:0]	R	Read back byte	8	Read 1 byte back is required for PXIe and RS232

4.2 Query Registers

These are requests for data registers, in that a request for certain data is made by writing to the specific register first, followed by reading back the requested data. Some registers may require instruction data to specify the type of data to return, while others do not need any. For example, the register GET_DEVICE_PARAM (0x30) returns the RF Frequency, IF1 Frequency, IF3 Frequency, etc.; the data returned depends on the parameter value of the instruction byte.

Returned data length is always 8 bytes (64 bits), with the first byte being the most significant (MSB). It is important that all 8 bytes are read to clear the interface buffers. The following will examine how the different interface buses handle the return data in the *Conversion Gain*

The data returned is the gain of the device, computed on board of the device. Note, the software API does not use this gain, it computes it on the host. Use of this register parameter along with bit [5] of register 0x1F provides convenience for embedded systems that do not use the API. The data received is float type and must be type casted back to it. For the u64_data received, the code in C/C++ would look something like:

```
uint32_t u32_data = (uint32_t)u64_data;

float conv_gain = *(float*)&u32_data;
```

Device Information Parameters and Format section.

Table 5. Query Registers

Register Name	Reg Add	Serial Range	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
GET_DEVICE_PARAM	0x30	[7:0]					Parameter			
GET_TEMPERATURE	0x31	[7:0]	Zeros [7:0]							
GET_DEVICE_STATUS	0x32	[7:0]	Zeros [7:0]							
GET_DEVICE_INFO	0x33	[7:0]					Info			
SERIAL_OUT_BUFFER	0x34	[55:0]	Zeros [55:0]							

4.2.1 Register 0x30 GET_DEVICE_PARAM

Write to this register to query the required device parameters from the device.

Bytes written	2
Bytes read	8

Bits	Type	Name	Width	Description
[3:0]	W	Parameter	1	0: Returns current RF frequency in Hz 1: Current IF frequency Hz 2: Current LO frequency Hz 3-8: Unused. 9: Reference DAC value 10: Convert Path Value 11: Attenuator values [7:0]: RF attenuation (LSB 0.5 dB) [15:8]: IF attenuation (LSB 0.25 dB) 12: Auto conversion parameters 13: Causes the device to configure and/or compute gain based on the conversion parameters, RF frequency, and IF frequency and return the gain. See bit [5] of register 0x1F.
[7:4]	W	Unused	4	Set to zeros
[63:0]	R	Read back bytes	64	Returned Data (See Device Parameter Data section for more info)

4.2.2 Register 0x31 GET_TEMPERATURE

Write to this register to query the device temperature.

Bytes written	2
Bytes read	8

Bits	Type	Name	Width	Description
[7:0]	W	Unused	7	Set to zeros
[31:0]	R	Temperature Data	32	These 32 bits of data need to be type casted back to float. i.e. float var= *(float*)&var_u32 where var_u32 is some unsigned integer that holds the 32 bits of read data.
[63:32]	R	Invalid data	32	Ignore

4.2.3 Register 0x32 GET_DEVICE_STATUS

Write to this register to query the current operating conditions.

Bytes written	2

Bytes read	8

Bits	Type	Name	Width	Description
[7:0]	W	Unused	7	Set to zeros
[0]	R	Pll status: LO1 SUM	1	The summing PLL of LO
[1]	R	Pll status: LO1 CRS	1	The coarse tuning PLL of LO
[2]	R	Pll status: LO1 FINE	1	The fine tuning PLL of LO
[3]	R	Pll status: CRS REF	1	Reference synth to the CRS synthesizer
[4]	R	Pll status: VCXO	1	100 MHz VCXO
[5]	R	Pll status: oCXO	1	0CXO, only valid when lock to external reference is enabled
[7:6]	R	Reserved	2	
[8]	R	LO Synth Standby	1	Internal LO synthesizer standby mode
[9]	R	Conversion Standby	1	Conversion path standby mode
[10]	R	Over Temp	1	If device temperature exceeds safe operating temp
[11]	R	Ext Ref enable	1	Lock to external reference enable/disable
[12]	R	Ext Ref detect	1	External reference signal detected bit
[13]	R	SynthLockMode	1	Integer-N loop or harmonic loop for locking
[14]	R	SynthLoopGain	1	Loop Gain set to (0) normal, (1) low
[15]	R	SpurSuppDisable	1	Automatically bounce between lock modes to minimize phase spurs
[16]	R	RefPortDirection	1	Ref port as input (0) or output (1)
[17]	R	PXI10M Enable	1	PXI backplane 10 MHz clock output enabled
[18]	R	Device_Accessed	1	Has software accessed (opened) the device

Bits	Type	Name	Width	Description
[63:19]	R	Invalid data	46	Ignore

4.2.4 Register 0x33 GET_DEVICE_INFO

Write to this register to query the device information, an example being the serial number.

Bytes written	2
Bytes read	8

Bits	Type	Name	Width	Description
[3:0]	W	DeviceInfo	4	0: Product serial number and interface type 1: Firmware and hardware revision 2: Calibration and manufacture date
[7:4]	W	Unused	4	Ignore
[63:0]	R	Data	64	Information data (see Device Info data)

4.2.5 Register 0x36 SERIAL_OUT_BUFFER

Writing to this register only provides the 64 clock edges (Reg + 7 data bytes) to transfer serial data from the device through SPI. Other interfaces do not use this register.

Bytes read	8

Bits	Type	Name	Width	Description
[55:0]	W	Unused	56	Zeros, just to provide clocking for SPI data from device
[63:0]	R	Data	64	8 bytes of data

4.2.6 Device Parameters Data and Format

The data read back from the GET_DEVICE_PARAM register has a total of 8 bytes, however, not all bytes contain valid data. The table below shows the valid data for each of the parameters.

Table 6. Device parameter data

Device Parameter Name	Param num	Data type	Byte 7	Byte 6	Byte 5	Byte 4	Byte 3	Byte 2	Byte 1	Byte 0

RF FREQ	0	U64	zeros	Data						
IF1 FREQ	1	U64	zeros	Data						
LO FREQ	2	U64	zeros	Data						
Reserved	3-8									
Reference DAC	9	U16	zeros	zeros	zeros	zeros	zeros	zeros	DAC value	
Signal Path	10	U16	zeros	zeros	zeros	zeros	zeros	zeros	Signal path	
Attenuators	11	U16							IF Atten	RF Atten
Auto conv params	12						RF level	Mixer level	IF level	L/A/H
Device conversion gain	13	F32					Gain in dB			

4.2.6.1 *Frequency parameter values for RF, IF, and LO*

Frequency parameter values are returned in the first 7 bytes with the last byte padded with zeros. The least significant bit (LSB) is 1/1000 Hertz (the data is returned in 1 one thousandth of a Hertz). Divide the data by 1000 to obtain the result in Hertz.

4.2.6.2 *DAC value*

Value is a returned as a 16-bit value

4.2.6.3 *Signal Path params*

The following table shows their returned values:

Bit	Description
[2:0]	Reserved
[3]	IF Sideband
[4:6]	Reserved
[7]	LO source select
[8]	Bypass conversion state
[9]	Reserved
[10]	RF Preamp state
[11]	If filter value bit 3
[12]	If filter value bit 2
[13]	If filter value bit 1
[14]	If filter value bit 0
[15]	IF output state

4.2.6.4 *Attenuator values*

Each attenuator value is returned as one byte, and the LSB is in 0.25 dB for the IF attenuator, and 0.5 dB for the RF attenuator.

4.2.6.5 *Auto conversion parameters*

Table 7. Auto conversion parameters

Bit	Description
[0]	SW auto calculation state
[1]	Auto Preamp control
[4:2]	Linearity mode
[5]	HW auto calculation state
[7:6]	Zero
[14:8]	IF level absolute value
[15]	IF level sign
[22:16]	Mixer level absolute value
[23]	Mixer level sign
[30:24]	RF level absolute value
[31]	RF level sign

4.2.6.6 *Conversion Gain*

The data returned is the gain of the device, computed on board of the device. Note, the software API does not use this gain, it computes it on the host. Use of this register parameter along with bit [5] of register 0x1F provides convenience for embedded systems that do not use the API. The data received is float type and must be type casted back to it. For the u64_data received, the code in C/C++ would look something like:

```
uint32_t u32_data = (uint32_t)u64_data;

float conv_gain = *(float*)&u32_data;
```

4.2.7 Device Information Parameters and Format

Not all 8 bytes read back contain valid data. The following table shows the valid bytes of data for each of the parameters.

Device Parameter Name	Param num	Data type	Byte 7	Byte 6	Byte 5	Byte 4	Byte 3	Byte 2	Byte 1	Byte 0
INTERFACE/SN	0	U32				Intrfce	Serial Number			
REVISIONS	1	F32	hardware revision				Firmware revision			
DATES	2	U32	Mfg year		Month	Day	Cal year		Month	Day

4.2.7.1 *Interface Information*

The first 4 bytes contain the device serial number as an unsigned 32-bit integer. Byte [4] contains the interface as represented in the following table.

Table 8. Interface ID

Bit	Description
[0]	PXI/PXIe (1 if available)
[1]	USB
[2]	SPI
[3]	RS232
[7:4]	Undefined

4.2.7.2 *Revision Information*

The first 4 bytes represent the hardware revision, and last 4 bytes represent the firmware revision of the device. These 4 bytes encompass a 32-bit floating point number so the data needs to be type casted from an unsigned 32-bit value to float value.

4.2.7.3 *Date Information*

The first 4 bytes represent the manufactured date, and the last 4 bytes represent the last calibration date. The date format is outlined in the following table.

Byte	Type	Description
[0]	U8	Day
[1]	U8	Month
[3:2]	U16	Year (i.e. 2016)

Section 2

Communication Interfaces and Calibration

5 Communication Interfaces

The SC5319A has a PXI express interface, while the SC5320A has 2 communication interfaces:

1. USB and SPI
2. USB and RS232

This section will examine the communication aspects of the product, focusing on data transfer to and from the device on each interface. Although the registers are identical for all interfaces, there are subtle differences in the implementation of the interfaces to transfer the data.

5.1 Communication Data Format

All data sent and received by all interfaces is sent as buffers of unsigned bytes. For example, to change RF frequency of the device to 32 GHz, the device performs the following:

1. Frequency is sent in 1000th of Hertz, so the data that represents the frequency is 32,000,000,000,000 milli-Hertz.
2. This number can be represented by a 64-bit unsigned long, and in Hexadecimal is 0x 0000 1D1A 94A2 000. Only the least 6 bytes are needed to represent all frequencies allowable for this device.
3. A byte data buffer needs to be 8 bytes for register RF_FREQUENCY (address 0x10), so the byte array buffer to be sent would be:

   [0x10][0x00][0x1D][0x1A][0x94][0xA2][0x00][0x00]

   The register address byte [0x10] is the first member of the buffer to be sent.

5.2 USB Interface

There are 2 transfer types for the USB interface.

- Control transfer
- Bulk transfer

5.2.1 Control Transfer

The USB control transfer parameters are:

ENDPOINT_IN	0x80
ENDPOINT_OUT	0x00
TYPE_VENDOR	0x40
RECIP_INTERFACE	0x01

5.2.2 Bulk Transfer

The USB bulk transfer parameters are:

ENDPOINT_IN	0x81
ENDPOINT_OUT	0x02

The bulk transfer from the host to the device operates on a loopback with a data buffer of 8 bytes. When a device register is addressed and the register task is completed, it will send back 8 bytes, which the host must read to clear the transfer buffers. Unlike the other interface methods, where only the required number of bytes needs to be sent for a given register, 8 bytes are needed for every USB bulk transfer. For example, if a configuration register requires only 4 bytes to be sent, these bytes will be the first of the 8 bytes and the last 4 bytes are zeros. The returned 8 bytes do not carry valid data for a configuration register. However, they do carry valid data for query registers.

5.3 SPI Interface

The SPI interface on the device is implemented using an 8-bit (single Byte) buffer for both the input and output, hence, it needs to be read and cleared by the device before consecutive bytes can be transferred to and from it. The process of clearing the SPI buffer and decisively moving it into the appropriate register takes CPU time, so a time delay is required between consecutive bytes written to or read from the device by the host. The chip-select pin ($\overline{CS}$) must be asserted low before data is clocked in or out of the product. Furthermore, pin $\overline{CS}$ must be asserted low for the entire duration of a register transfer.

Once a full transfer has been received, the device will proceed to process the command and de-assert low the SRDY pin. The status of this pin may be monitored by the host because when it is de-asserted low, the device will ignore any incoming data. The device SPI is ready when the previous command is fully processed and the SRDY pin is re-asserted high. It is important that the host either monitors the SRDY pin or waits for 500 μs between register writes.

There are 2 SPI modes: 0 and 1. The default mode is 1 (C_polarity = 0, C_phase = 1), where data is clocked in and out of the device on the falling edge of the clock signal. In mode 0 (C_polarity = 0, C_phase = 0), data is clocked in and out on the rising edge. To select mode 0, pin 8 of the interface connector must be pulled low to ground as the device is powered on or as the reset line (pin 12) is toggled low-high. If pin 8 is pulled high or left unconnected, mode 1 is selected.

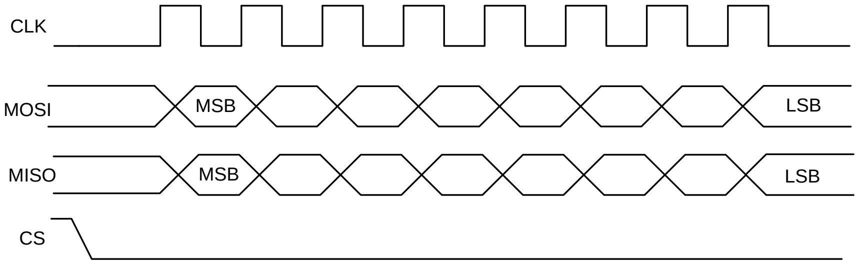

Figure 9. SPI Mode 1 shown.

Register writes are accomplished in a single write operation. Register lengths vary, depending on the register. They vary in lengths of 2 to 8 bytes, with the first byte sent being the register address followed by the data associated with that register. The ($\overline{CS}$) pin must be asserted low for a minimum period of 1 μs (T_S, see *Figure 10*) before data is clocked in, and must remain low for the entire register write. The clock rate may be as high as 2.0 MHz (T_C = 0.5 μs), however, if the external SPI signals do not have sufficient integrity due to trace issues, the rate should be lowered.

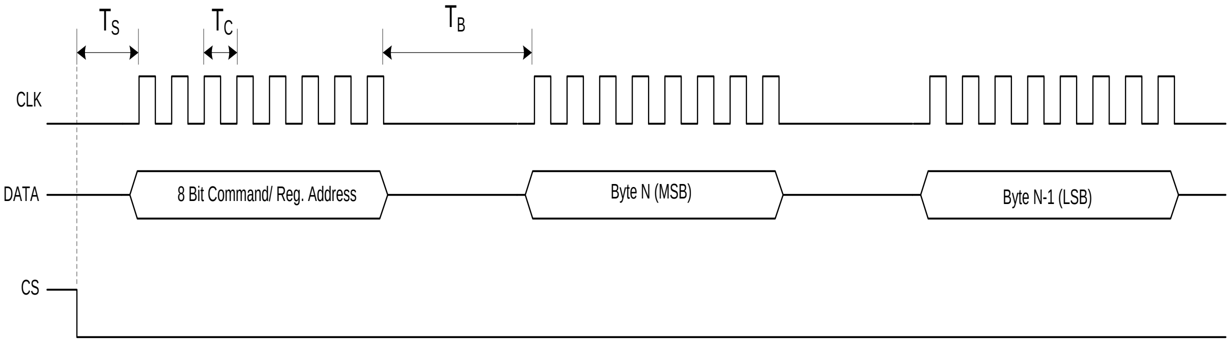

Figure 10. SPI timing.

As mentioned above, the SPI architecture limits the byte rate since after every byte transfer the input and output SPI buffers need to be cleared and loaded respectively by the device SPI engine. Data is transferred between the input buffer and internal register buffers. The time required to perform this task is indicated by T_B, which is the time interval between the end of one byte transfer and the beginning of another. The recommended minimum time delay for T_B is 5 μs. The number of bytes transferred depends on the register. It is important that the correct number of bytes is transferred for the associated register, because once the first byte (MSB) containing the device register address is received, the device will wait for the desired number of associated data bytes. The device will hang if an insufficient number of bytes are written to the register. To clear a hung condition, the device will need an external hard reset. The time required to process a command is also dependent on the command itself. Measured times for command completions are typically between 50 μs to 300 μs after reception.

5.3.1 Writing the SPI Bus

The SPI transfer size (in bytes) depends on the register being targeted. The first byte sent is the register address and subsequent bytes contain the data associated with the register. As data from the host is being transferred to the device via the MOSI line, data present on its SPI output buffer is simultaneously transferred back, MSB first, via the MISO line. The data returned is invalid for configuration registers. The following figure shows the contents of a single 3-byte SPI command written to the device. The Hardware Registers section provides information on the number of data bytes and their contents for an associated register. There is a minimum of 1 data byte for each register even if the data contents are "zeros".

23								15								7							0
	Register Address									Byte 1								Byte 0					

Figure 11. Example of a 3-byte write

5.3.2 Reading the SPI Bus

Data is simultaneously read back during an SPI transfer cycle. Requested data from a prior command is available on the device SPI output buffers, and these are transferred back to the user host via the MISO pin. Obtaining valid requested data would require querying the SERIAL_OUT_BUFFER, which requires 8 bytes or 64 clock cycles; 1 byte for the device register

(0x36) and 7 empty bytes (MOSI) to clock out the returned data (MISO). An example of reading the device RF parameters (IF3 frequency) from the device is shown in the following figure.

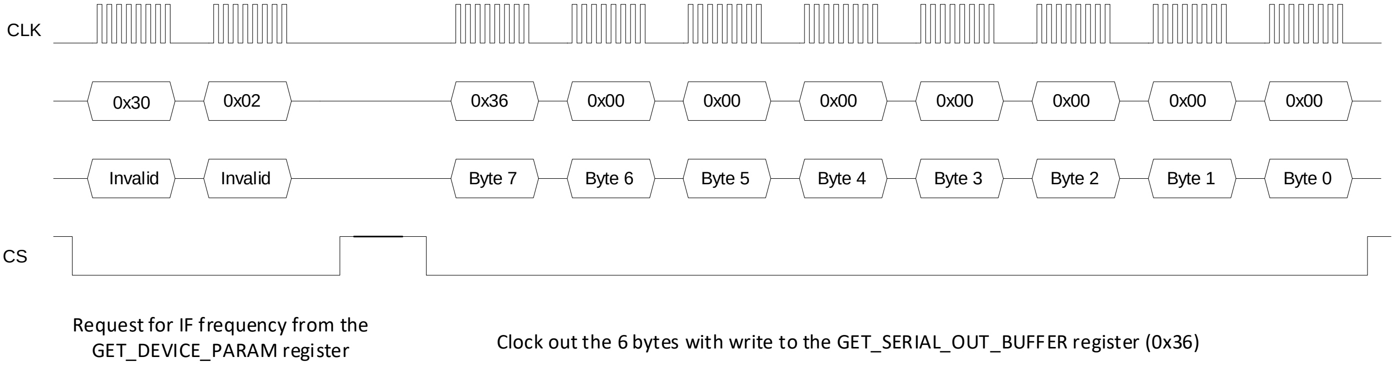
CLK
0x30
0x02
0x36
0x00
0x00
0x00
0x00
0x00
0x00
0x00
Invalid
Invalid
Byte 7
Byte 6
Byte 5
Byte 4
Byte 3
Byte 2
Byte 1
Byte 0
CS
Request for IF frequency from the GET_DEVICE_PARAM register
Clock out the 6 bytes with write to the GET_SERIAL_OUT_BUFFER register (0x36)

Figure 12. Query example: Write followed by Read to the GET_SERIAL_BUFFER

In the figure above, the first transfer cycle makes the request for IF3 frequency data through the GET_DEVICE_PARAM register. The subsequent cycle is to clock the data that was requested by sending 64 clocks into the GET_SERIAL_BUFFER register.

5.4 RS232 Interface

The RS232 version of the SC5320A has a standard interface buffered by an RS232 transceiver so that it may interface directly with many host devices, such as a desktop computer. The interface connector for RS232 communication is labeled "Digital I/O" on the front of the panel. Refer to Table 3 for position and pin-out information. The RS232 device communication control set is provided in the following table.

Table 9. RS232 Control Setting

Baud rate	Rate of transmission
Data bits	The number of bits in the data is fixed at 8.
Parity	Parity is 0 (zero)
Stop bits	1 stop bit
Flow control	0 (zero) or none

Only 3-wire RS232 is required since hardware flow control is not used. These connections are the Tx, Rx, and Gnd. This interface is common on most host computers and microcontrollers, so user access to host ports is readily provided by the computer OS or microcontroller hardware registers.

5.4.1 Writing to the Device Via RS232

It is important that all necessary bytes associated with any one register are fully sent. In other words, if a register requires a total of 6 bytes (address plus data), then all 6 bytes must be sent even though the last byte may be null. The device, upon receiving the first register addressing byte, will wait for all the associated data bytes before acting on the register instruction. Failure to complete the register transmission will cause the device to behave erratically or hang. Information for writing to the configuration registers is provided in *Table 4. Configuration Registers*. Upon the execution of the register that was sent, the device will return one (1) byte of data with bit 1 high to indicate success. This byte must be read by the host to clear its received buffer so that reading subsequent registers will not contain corrupted data. Furthermore, reading back this byte will ensure that the device is ready for the next register command.

5.4.2 Reading from the Device Via RS232

To query information from the device, the query registers are addressed, and data is returned. *Table 5. Query Registers* contains the query register information. As with the configuration registers, it is important that all data byte(s) (write) associated with the query registers are sent even if they are null. All queries will return 8 bytes of data (read) with the first received byte being the most significant (MSB). Section *4.2 Query Registers* provides the format details of the received data.

5.5 PXI Express

The PXIe interface contains a high-speed PCIe-to-Serial bridge chip. This bridge chip communicates with the onboard microcontroller serially. The interface on the bridge chip resides at offset addresses between 0x00 and 0xFF from BAR0; which is memory mapped. A kernel level driver for the operating system is needed to access this memory address. A simple driver using IO controls should be sufficient to read and write byte data to this block of addresses. Although SignalCore provides the driver and API for these products, information is provided here for users who may need to write drivers for a different operating system or a different driver. An example would be writing the API for the Linux operating system.

5.5.1 Setting Up the PCI to Serial Bridge

The serial function of the bridge chip must first be initialized before it can communicate with the onboard microcontroller, and hence establish communication between the microcontroller and the PXIe bus. The initialization can be done at the kernel level mode or at the user level mode, the decision is left to the user. The following table lists the programing order of the bridge register addresses to initialize and setup the serial port function.

Step	BAR0 Register Address	Data (Byte)
1	0x88	0x01
2	0x04	0x00
3	0x03	0x80
4	0x00	0x07
5	0x01	0x00
6	0x03	0x0D
7	0x02	0x07

5.5.2 Writing to the Device

Bytes that are written to the device must go through the bridge chip. This section will examine the write cycle of each byte, and then the write cycle of each device register. Do note the difference between the bridge register addresses and the device register addresses.

5.5.2.1 *Single Byte Write*

The serial transfer buffer register address is located at 0x00 offset from BAR0 of the bridge chip, however, before writing byte data to this register, its status needs to be checked to confirm that it is ready to accept a new buffer of bytes. The status register is located at 0x05; it must be read and bit 7 must be **high** to indicate that the transfer register is ready to receive the next byte buffer. Checking the status register of the serial bridge chip is required before every new command write.

5.5.2.2 *Device Register Write*

The process of writing the device registers is the same as writing to an RS232 port, so the description of Section *5.4.1 Writing to the Device Via RS232* is applicable. Writing the device registers involves sending byte-by-byte data as described previously. Section *4.1 Configuration Registers* provides information on the number of configuration-write bytes needed for each device register. The first byte sent is the device register address, followed by the most significant byte of the register's associated data. When a device register is fully written, that is, all its data has been sent to the device, it will return 1 byte. This returned byte must be read (by the host) to clear the transfer buffer so that later received data are not corrupted. Section *5.5.3.1 Single Byte Read* describes how a byte read cycle is performed.

5.5.3 Reading from the Device

Device data is passed back to the host via the bridge chip byte-by-byte. The following sections discuss a single byte read process and an entire register read process.

5.5.3.1 *Single Byte Read*

The serial transfer buffer register address is located at 0x00 offset from BAR0 of the bridge chip. Before valid data can be read from the transfer register, its ready status must first be confirmed. The status register is located at 0x05; it must be read and bit 0 must be **high** to indicate that valid data is available. Checking the status register for available data is required before **every byte** is read.

5.5.3.2 *Device Register Read*

After a write request to the device is made, 8 bytes of data is available to be read back. Use the single byte read process, as mentioned previously, to read all the bytes. See Section 4.2 for information of the exact number of request write bytes and the number of request read bytes, which is 8. All 8 bytes must be read to fully clear the transfer buffer. The first byte read is the most significant byte.

6 Calibration

6.1 Calibration EEPROM Map

Table 10 represents the EEPROM map of the device calibration values. All values are stored as little-endian 4-byte floating point numbers. Every point is 4-bytes long. Access to the data is possible through the CAL_EEPROM_READ register, which reads 8 bytes starting at the address pointed to by the register input. The absolute address of the EEPROM at which calibration data is stored is 0x145A.

Table 10. Calibration EEPROM Map

Offset address	Points	Length (Bytes)	Data Type	Description
0x00	664	664	U_8	Factory Reserved
0x298	1	4	Float_32	Calibration Temperature
0x29C	6	12	Float_32	Temp coeffs, band1 = <13 GHz, <13Ghz, band2 < 20GHz, and band3 > 20 GHz. (c1, c2, c1, c2, c1, c2)
0x398	50	200	Float_32	IF response cal frequencies 100, 200, 300, ..., 3500 in MHz
0x4B0	50	200	Float_32	IF3_response, relative gain to 1 GHz at each IF cal frequency
0x5C8	30	120	Float_32	Relative IF atten 1 dB.....30 dB, 1 dB step
0x6B8	161	644	Float_32	Bypass response cal frequencies 250, 500, 750, ..., 26000 in MHz
0x93C	161	644	Float_32	Bypass absolute gain at each bypass cal freq
0xBC0	85	340	Float_32	RF cal frequencies, 6 GHz to 26.5 GHz, 250 MHz step
0xDA0	85	340	Float_32	RF upper sideband abs gain @ each rf cal freq
0xF80	85	340	Float_32	RF lower sideband abs gain @ each rf cal freq
0x1160	85	340	Float_32	Rf rel amplifier gain @ each rf cal freq
0x1340	2635	10540	Float_32	RF atten, 1-30 dB, 1 dB step for each freq a1_f0, a1_f1,...,a1_fN a2_f0, a2_f1,...,a2_fN a30_f0, a30_f1,...,a30_fN

6.2 Conversion Gain

It is difficult to make gain measurements of every configuration setting of the downconverter as there are many combinations of attenuator and frequency settings, in addition to the various operating temperatures. However, the gain of the device can be computed to a reasonable level of accuracy with less data if the assumption is that every component setting is independent from one another. For example, it is assumed that the relative (to center of 2 GHz) frequency response of the IF_ATTEN attenuator is the same for all attenuation settings. Another assumption made, which may introduce the most error, is the IF response is the same for all LO and RF values. As an example, the relative IF value at 2.5 GHz is the same for the sets (LO = 12.5 GHz, RF = 10 GHz) and (LO = 16.5 GHz, RF = 14 GHz). Any change in gain is assumed to have occurred only in the RF path and not the IF path.

6.3 Absolute Gain of the RF Conversion Path

An absolute reference state of the device must be determined first, assuming independency, and the ability to compute the gain of the downconverter at any setting,. Once this reference state is established, compensation can be applied to other relative configured states of the device. The established reference state of the device is as follows:

- All attenuators are set to 0 dB
- IF value is set at 2 GHz
- RF preamplifier is disabled.
- RF is tuned from 20000 MHz to 40000 MHz in 250 MHz steps

The gain of the device in this state is measured at different frequencies in the range of 20000 MHz to 40000 MHz. This set of measurements is the RF absolute gain stored at starting relative address 0xBC0 of the calibration EEPROM. All other measurements taken as deviations from this reference setting are relative.

6.4 Gain Through the Bypass RF Conversion Path

The absolute gain through the bypass path is stored beginning at address 0x6B8. The absolute gain measurement is made every 250 MHz, between 250 MHz and 40000 MHz.

6.4.1 Applying Calibration

The gain and attenuator values are relative measurements from the absolute reference values made over various RF frequencies. These relative values are either subtracted (attenuation) or added (gain) to the absolute gain value to determine the gain of the relative configuration. For example, assume the RF_ATTEN is at 5 dB, IF3_ATTEN is at 2 dB, IF frequency set to 1550 MHz, and the RF frequency is set to 28.1 GHz. A possible systematic approach would be:

1. Compute the absolute gain value by interpolation because there is no measurement value at 28.1 GHz. A simple linear interpolation between the measured 28.0 GHz and 28.25 GHz values would provide a good estimate. A local spline interpolation over 5-6 surrounding points would provide better accuracy. Let us call this value $G(rf)$.

2. Compute the relative RF attenuator value by interpolation because there is no measurement value at 28.1 GHz. A simple linear interpolation between the measured 28.0 GHz and 28.25 GHz values would provide a good estimate. A local spline interpolation over 5-6 surrounding points would provide better accuracy. Let us call this value $A(5dB, rf)_{rf\_atten}$.

3. The IF Attenuator value is read directly from memory as $A(2dB)_{if\_atten}$, however it must be corrected for frequency offset from 2000 MHz. There are offset gain response values measured at 1500 MHz and 1600 MHz, so a simple linear interpolation between 2 points should be sufficient. Call this correct gain value $G(if)$.

4. The gain for this configuration is calculated using:

   $$G = G(rf)_{abs} + G(if) - A(5dB, rf)_{rf\_atten} - A(2dB)_{if\_atten}$$

   If the current device temperature is different from the calibration temperature, the gain correction due to temperature difference is computed using

   $$\Delta G_{temp} = C(b1)_1 (T - T_0) + C(b1)_2 (T - T_0)^2 \ .$$

   ΔG_{temp} is the gain correction, $C(b1)_1$ and $C(b1)_2$ are the first and second order temperature gain coefficients respectively for RF band 1, and T and T_0 are the current temperature and calibration temperature respectively. Adding this correction to the previously calculated gain will compensate the value for temperature deviation.

   $$G = G(rf)_{abs} + G(if) - A(5dB, rf)_{rf_{atten}} - A(2dB)_{if_{atten}} + \Delta G_{temp}$$

Revision Table

Revision	Revision Date	Description
0.1	10/19/21	Document Created, preliminary
1.0	03/24/22	Release
1.1	09/26/22	i. Corrected byte assignments for register 0x1F, AUTO_CONV_PARAMS. ii. Added parameter (0x1D) 13 to register 0x30. iii. Added description of the SC5319A PXIe front panel.
1.2	12/7/22	Corrected torque range
1.3	07/10/23	Corrected for bypass frequency range