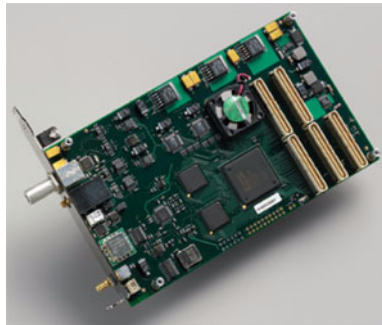


PCI SS/GS SRXL

SRXL Satellite Tuner
for use with PCI SS/GS Main Board



April 2006
008-02663-00



The information in this document is subject to change without notice and does not represent a commitment on the part of Engineering Design Team, Inc. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of the agreement.

Engineering Design Team, Inc. ("EDT"), makes no warranties, express or implied, including without limitation the implied warranties of merchantability and fitness for a particular purpose, regarding the software described in this document ("the software"). EDT does not warrant, guarantee, or make any representations regarding the use or the results of the use of the software in terms of its correctness, accuracy, reliability, currentness, or otherwise. The entire risk as to the results and performance of the software is assumed by you. The exclusion of implied warranties is not permitted by some jurisdictions. The above exclusion may not apply to you.

In no event will EDT, its directors, officers, employees, or agents be liable to you for any consequential, incidental, or indirect damages (including damages for loss of business profits, business interruption, loss of business information, and the like) arising out of the use or inability to use the software even if EDT has been advised of the possibility of such damages. Because some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitations may not apply to you. EDT's liability to you for actual damages for any cause whatsoever, and regardless of the form of the action (whether in contract, tort [including negligence], product liability or otherwise), will be limited to \$50 (fifty U.S. dollars).

No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, without the express written agreement of Engineering Design Team, Inc.

Copyright © 2006 Engineering Design Team, Inc. All rights reserved.

EDT and Engineering Design Team are trademarks of Engineering Design Team, Inc.

Xilinx is a registered trademark of Xilinx, Inc.

Contents

Installation and Setup	2
Included Files	2
Loading the Bitfiles	3
Building Applications	3
Running the Example Application	3
Accessing the SRXL Devices	3
Serial Control Register	4
Getting the Board ID	4
Setting the Reference Clock	4
Setting the Sample Clock	5
Programming the Sample Clock	5
Resetting the Sample Clock	6
Sample Clock Control Register	6
DDS Control Register	6
DDS Data Register	7
Analog-to-Digital Converters	7
Setting the IF Tuner	7
Programming the PLL Synthesizer	7
PLL Low Register	9
PLL Mid Register	9
PLL High Register	9
Setting the L-band Tuner	9
Writing the Registers	10
Reading Status	10
Setting the Baseband Gain	10
Setting the Baseband Lowpass Filter Cutoff Frequency	10
Setting the Receive Frequency	11
Maxim Address Register	13
Maxim Data Register (write)	13
Maxim Data Register (read)	13
Setting the Gain	13
DAC A Low Register	15
DAC A High Register	15
DAC B Low Register	15
DAC B High Register	16
Acquiring a Signal	16
Capture Control Register (read)	16
Capture Control Register (write)	17
Read Data Register	17
DMA Mode	18
Test Mode	18

Accessing the Graychips	18
Graychip Address Register	19
Graychip Data Register.....	19
Pinouts.....	20
References	33

Programming the SRXL Satellite Tuner

The SRXL Satellite Tuner is a single-slot PCI Bus mezzanine board for use with the PCI SS/GS main board enabling you to capture an L-band or intermediate frequency (IF) analog signal and send it to a host computer for processing. The SRXL can acquire 12-bit digital samples in parallel from each of four signals: IF I and Q components, and L-band I and Q components.

The SRXL includes the following components:

- a 10 MHz reference clock,
- an IF tuner,
- an L-band tuner,
- dual variable-gain amplifiers for setting signal gain,
- two 12-bit analog-to-digital converters to digitize the incoming signal,
- a programmable-frequency sample clock synthesizer to set the sample rate,
- a Xilinx[®] field-programmable gate array for running custom applications, and
- two Graychips for additional signal processing.

Samples are 32 bits wide. Sample data for each signal is stored in a FIFO with a capacity of 8 KB — 2 KB of samples. You can access this data in either DMA mode or test mode.

- In *DMA mode*, you can transfer IF data on DMA channel 0 and L-Band on DMA channel 1. Data is structured in 32-bit words that contain the I component in bits 11–0 and the Q component in bits 29–16. Software determines whether both DMA channels are simultaneously active, or just one operates at a time.
- Alternatively, *test mode* allows sample data to be read out of the FIFO one byte at a time using the IO register interface.

Installation and Setup

To install the SRXL mezzanine board:

1. Install the Pcd driver software as specified on the software disk jacket.
2. Power down the host computer.
3. If necessary, attach the SRXL mezzanine board to the main PCI SS/GS board using the five 64-pin connectors (shown in [Figure 1](#)) and the four posts, one in each corner.
4. Fit the board assembly into the PCI Bus connector in the host computer as specified on the software CD jacket.
5. Power the host back on.

Included Files

The SRXL ships with the following SRXL-specific software:

<code>srxl_top.bit</code>	The VHDL bitfile for the Xilinx on the SRXL mezzanine board.
<code>srxl_4io.bit</code>	The VHDL bitfile for the user interface Xilinx on the main PCI SS/GS board.
<code>srxlload</code>	A C application that loads the bitfiles specified above into the SRXL Xilinx and the user interface Xilinx on the main board.
<code>srxlload.c</code>	The C source for the application above.
<code>srxl_debugger</code>	A C example application that exercises most of the functionality available on the SRXL and provides several debugging aids.
<code>srxl_debugger.c</code>	The C source for the application above.
<code>srxl_gc_diag</code>	A C diagnostic application to test the Graychips.
<code>srxl_gc_diag.c</code>	The C source for the application above.
<code>fft</code>	A C example application to perform a Fast Fourier Transform on both real and imaginary components of the captured data.
<code>fft.c</code>	The C source for the application above.
<code>halfbandfft</code>	A C example application to perform a Fast Fourier Transform on both real and imaginary components of the captured data and then performs image rejection to minimize the reflected signal as best it can.
<code>halfbandfft.c</code>	The C source for the application above.
<code>lib_srxl.c</code>	C source for routines called by the example applications. These routines in turn call routines in <code>libedt.c</code> , the EDT DMA Library .
<code>lib_srxl.h</code>	A header file for the C source routines.

NOTE The SRXL requires that the main PCI SS/GS board flash PROM is loaded with 4-channel firmware; single- or 16-channel firmware is not compatible with the SRXL bitfile.

Loading the Bitfiles

NOTE In the instructions below, placeholders such as *unit number* appear in italics. Replace them with the appropriate values.

To load the bitfiles and get the SRXL ready to use:

1. Using a command window, run `pciload` to determine the unit number of the PCI SS/GS main board — by default, 0 — and to verify:
 - that the host detects your PCI SS/GS main board, and
 - that it's loaded with 4-channel firmware.
2. If the PROM ID includes the string `pcigs4` or `pciss4`, then 4-channel firmware is already loaded. If not, load the 4-channel firmware with the appropriate command, either:

```
pciload -u unit number pciss4 or pciload -u unit number pcigs4
```

3. At the prompt, press **Enter** to confirm the loading operation.
4. Load the SRXL bitfile `srxl_top.bit` and the user interface Xilinx bitfile `srxl_4io.bit`, if necessary:

```
srxlload -u unit number srxl_top.bit
```

The onscreen feedback verifies that the bitfiles are loaded and the LED blinks rapidly.

NOTE You can substitute your own VHDL bitfile for `srxl_top.bit` if necessary.

Building Applications

Executable files and Pcd source files are at the top level of the EDT distribution directory. Run `make` at the top level, therefore.

SRXL source code files are in the `./srxl` subdirectory.

Running the Example Application

The example application `srxl_debugger` provides a way to:

- reload the SRXL Xilinx with the bitfile `srxl_top.bit`,
- reinitialize the SRXL board to the initial values of the `srxl_top.bit` firmware,
- run a diagnostic checksum test on the Graychips,
- read and write the SRXL and user interface Xilinx registers,
- access and set values on the SRXL devices, and
- perform complex Fast Fourier Transforms (FFT) on the captured data.

To run the example application, at the Pcd Utilities prompt, enter:

```
srxl_debugger
```

At the SRXL Debugger prompt, enter `h` for a list of all the usage options and their descriptions.

Accessing the SRXL Devices

Your application can access the sample clock synthesizer, IF and L-band tuners, the Graychips, and the gain adjustments for each signal by reading and writing the appropriate registers in the SRXL

Xilinx. Firmware then serializes your instructions and sends the resulting data or instructions to the target device. These serial transfers are slower than the register programming interface. To avoid ignored or corrupted communications, before starting a new register access, check bit 7, the XFER_BUSY bit, in the Serial Control Register to determine that all current or pending serial transfers are complete.

NOTE Wait for XFER_BUSY to be low before starting any register access intended for the SRXL devices.

Serial Control Register

Size	8-bit
I/O	read-write
Address	0x54
Access	SERIAL_CTRL
Comment	Control and status bits for functions accessed by serial interface

Bit	Name	Description
7	XFER_BUSY	Firmware sets this when any of the serial interfaces are busy. Wait till this bit is clear before starting any new read or write to any of the SRXL devices.
6	I2C_ERROR	Firmware sets this after any I ² C data transfer for which an ACK was not received, indicating that the Maxim2118 is not responding. Firmware clears before starting a new I ² C data transfer.
5	IF_FTEST	A status bit from the IF local oscillator PLL synthesizer. By default, a value of 1 indicates the PLL loop is locked; when zero, that it is not locked.
4–1		unused
0	I2C_READ	Set this when reading the Maxim2118 Status Register. Clear this when writing to any Maxim2118 register.

Getting the Board ID

The SRXL uses an extended board ID. To get the board ID and revision information, after you've installed the board and loaded the firmware, enter:

```
extbdid
```

Setting the Reference Clock

The SRXL requires a reference clock signal of 10 MHz. Software can select between the onboard crystal oscillator or an external reference input. The reference clock output drives the sample clock synthesizer, the IF local oscillator PLL, and the L-band tuner; it's also available to the Xilinx. Changing the reference clock source can affect any of these devices.

By default, the SRXL uses the signal from the onboard device.

To specify an external reference input as the signal source, set bit 7 of the Sample Clock Control Register to one. If you do so, ensure that a 10 MHz signal is coming in the SMB connector. (See [Figure 1.](#))

Clear bit 7 to select the onboard crystal oscillator as the reference clock signal. See [Sample Clock Control Register on page 6](#).

Setting the Sample Clock

The A/D sample clock is generated by an Analog Devices AD9951 direct digital synthesizer (DDS). The DDS output frequency range is 1–65 MHz. The AD9951 specifies a DDS output maximum of 65 MHz; output is ordinarily 1–60 MHz.

The AD9951 receives a 10 MHz reference clock, either generated onboard, or from the external reference clock source. This clock is used directly or multiplied 4–20 times inside the AD9951 for the DDS sample clock rate which, with the 32-bit tuning frequency word, determine the output frequency. (The tuning word is described on [page 5](#).) A differential analog lowpass filter with a cutoff of 20 MHz removes alias components from the synthesized clock output.

The differential DDS output clock is an input to the Xilinx, which sends it to the A/D converters and, optionally, the Graychips, as specified by the application.

Programming the Sample Clock

The A/D sample clock is specified by programming six internal registers of the AD9951 direct digital synthesizer (DDS). The size of each register varies, but all are programmed by writing the most significant byte first.

Register descriptions for the [Analog Devices AD9951](#).

To write to the DDS registers:

1. Write the address of the DDS target register to the DDS Data register.
2. Monitor the XFER_BUSY bit in the Serial Control Register until you see it's clear.
3. Write all the bytes in that register, starting with the most significant. Monitor the XFER_BUSY bit after writing each byte and before writing the next, to ensure that one transfer is complete before starting the next.
4. Repeat step 1 through step 3 as needed to write as many DDS registers as required.
5. Issue a DDS update by toggling bit 0, the update bit, in the DDS Control register.

The DDS Control and Data registers are described on [page 6](#).

The DDS output frequency is:

$$\text{Ref Freq} * \text{REFCLK_multiplier} * \text{Tuning Word} / 2^{32}$$

The reference frequency is 10 MHz, so the tuning word is:

$$\text{Tuning Word} = (2^{32} * \text{DDS Output Frequency}) / (10 * \text{REFCLK_multiplier})$$

The tuning word is 32 bits unsigned. Whenever the SRXL firmware is loaded, the firmware programs the DDS device with a reference clock multiplier of 20 and a tuning word value that sets the sample clock frequency to 15 MHz. All other bits are set to their default value of 0.

Resetting the Sample Clock

You can reset the sample clock synthesizer by toggling bit 1, the reset bit, in the DDS Control register — first set, then clear, this bit. You must then reprogram all the AD9951 registers for the sample clock to operate.

The registers that control the sample clock and the A/D converter output are discussed below.

Sample Clock Control Register

Size	8-bit
I/O	read-write
Address	0x5F
Access	CLK_CTRL

Bit	Name	Description
7	EXT_REF_SEL	Setting this bit selects the external reference input as the source for the 10 MHz reference clock. For proper operation, ensure that a 10 MHz clock signal source is connected to the SRXL. When clear, selects the onboard TCXO for the 10 MHz reference clock.
6	LED_ON	Setting this bit sets the LED to a rapid blink. Clearing it sets the LED to a slow blink.
5	AD_DFS	Setting this bit selects twos-complement output format for the A/D converters. When clear, selects offset-binary format.
4	AD_DCS	The analog-to-digital clock stabilizer bit. Set high for sample clocks of 40 MHz and above. Set low for frequencies below 40 MHz.
3–0		not used

NOTE Not only the sample clock synthesizer, but also the IF and L-band tuners, the local oscillator, and the Xilinx use the external clock source specified by bit 7, EXT_REF_SEL. Changing the reference clock source can affect any of these devices.

DDS Control Register

Size	8-bit
I/O	read-write
Address	0x50
Access	DDS_CTRL

Bit	Name	Description
7-2		not used
1	DDS_RESET	Toggling (first setting, then clearing) this bit resets the AD9951 device, which then needs to be reprogrammed before operation.
0	DDS_UPDATE	Setting this bit transfers all bytes written to its internal buffers to its internal registers.

DDS Data Register

Size	8-bit
I/O	read-write
Address	0x51
Access	DDS_DATA
Comment	The DDS Data register holds the values to written to the AD9951. Though this register is read-write, it is ordinarily used only writing; reading this register yields the last value written to it.

Analog-to-Digital Converters

The analog signal is sampled at the rate determined by the sample clock rate and digitized using the two 12-bit analog-to-digital converters on the AD9238 direct digital synthesizer, one for the IF signal and one for the L-band signal.

The maximum sample rate is 65 MHz. For sampling rates of 40 MHz and above, set the Duty Clock Stabilizer bit — bit 4 of the [Sample Clock Control Register](#).

If either input signal exceeds the range of the analog-to-digital converter, the overrange is reported in bit 4 of the [Capture Control Register \(read\)](#) for the L-band signal, or bit 5 for the IF signal. Read those bits to determine if an overrange occurred since the last reset of the FIFO.

You can select the data output format using the Data Format Select bit — bit 5 of the [Sample Clock Control Register](#).

I and Q data are multiplexed onto one 12-bit output, so the data rate is twice the sampling rate. Twelve bits of I data are output on the rising edge of the sample clock, and 12 bits of Q data on its falling edge. This digitized data is an input to the SRXL Xilinx.

Setting the IF Tuner

The IF channel decoder uses a Linear Tech LT5546 that requires a local oscillator input at twice the desired receive center frequency. The local oscillator is generated using a 250–450 MHz VCO with a National LMX2364 PLL synthesizer, which uses the 10 MHz reference clock signal. For local oscillator frequencies below 250 MHz, you can enable a divide-by-two prescaler.

The LMX2364 was chosen for its performance as an IF synthesizer; the SRXL does not use all its capabilities, so certain register bits are left programmed in a default state.

Programming the PLL Synthesizer

The LMX2364 PLL chip has seven programmable internal registers, accessed by writing a 24-bit PLL programming word into the byte-wide registers PLL_LOW, PLL_MID, and PLL_HIGH (described starting on [page 6](#)). After the PLL_HIGH byte is written, the Xilinx transfer the 24-bit value serially to the LMX2364.

Ordinarily, you'll need to change only the output frequency, which is programmed in bits 3–19 of internal PLL Register 1, as shown in [Table 1](#):

Table 1. PLL Register 1

PLL Register 1 Bits	Value
23–20	0000
19–3	Frequency Word
2–0	001

The Frequency Word is set to half the desired VCO frequency in MHz: legal values are in the range of 125–225 MHz, so only the eight least significant bits of the 17-bit word are actually used. To assure proper operation, make sure the upper 9 bits are zero.

For IF receive frequencies of 125 MHz and above, the Frequency Word is set to the desired receive frequency in MHz. Disable the prescaler by clearing the LO_DIV_SEL (bit 4 in PLL Register 2).

For IF receive frequencies from 63–112 MHz, set the Frequency Word to twice the desired receive frequency in MHz, and enable the divide-by-two prescaler by setting LO_DIV_SEL (bit 4 in PLL Register 2) to high.

NOTE For frequencies of 113–124 MHz, operation is not characterized.

After writing the frequency word in PLL Register 1 and LO_DIV_SEL in PLL Register 2, allow 100 ms for the PLL to stabilize before reading bit 5 of the [Serial Control Register](#) to determine if the PLL is locked.

The prescaler is set using bit 1 of PLL Register 2, as shown in [Table 2](#):

Table 2. PLL Register 2

PLL Register 1 Bits	Name	Value
23–5		0000000000000000
4	LO_DIV_SEL	0 for Receive Frequency of 120–225 MHz 1 for Receive Frequency of 63–119 MHz
3–0		1010

NOTE The divide-by-two prescaler, LO_DIV_SEL, is brought out of the PLL on the FloutIF pin, used in this case as a programmable IO pin.

When you load the firmware, the seven PLL registers are reset as shown in [Table 3](#):

Table 3. LMX2364 PLL Register Reset

PLL Register	Value (0x)	Description
0	0C0024	Set R divider to 5 for fcomp 2 MHz, charge pump gain .8, positive PD out
1	000489	Set N counter to 145. for 145/72.5 MHz IF, depending on whether the divide-by-two prescaler is enabled (setting of LO_DIV_SEL).
2	00000A	FastLock disabled, FLOUTIF (used as LO_DIV_SEL) pin set low.
3	07FFFB	Reasonable values for RF synthesis, although disabled.
4	FFFFFC	Powerdown RF synth half of PLL.
5	00000D	FLOUTRF pin held low.
6	000046	Digital lock detect on IF synthesis to Xilinx.

PLL Low Register

Size	8-bit
I/O	read-write
Address	0x5C
Access	PLL_LOW

Bit	Name	Description
7–3	PLL_LOW	Bits 7–3 of the PLL internal register.
2–0	LMX_REG_ADDR	The register address in the LMX2364

NOTE The SRXL leaves many PLL register bits programmed in their default state.

PLL Mid Register

Size	8-bit
I/O	read-write
Address	0x5D
Access	PLL_MID

Bit	Name	Description
7–0	PLL_MID	Bits 15–8 of the PLL Internal register.

PLL High Register

Size	8-bit
I/O	read-write
Address	0x5E
Access	PLL_HIGH

Bit	Name	Description
7–0	PLL_HIGH	Bits 23–16 of the PLL Internal register.

[Product information for the LMX2364 PLL.](#)

Setting the L-band Tuner

The Maxim 2118 L-band decoder contains six programmable registers to control the receive frequency, baseband bandwidth and gain. You can access these registers using three SRXL registers: the Serial Control Register SERIAL_CTRL_REG, the Maxim Address Register I2C_SLAVE_REG_REG, and the Maxim Data Register I2C_SLAVE_DAT_REG. See [Serial Control Register on page 4](#) for details of the Serial Control Register; the others are described below.

NOTE The Maxim 2118 uses an I²C bus, which appears as I2C in register and bit names and in code.

The firmware initializes the six Maxim 2118 registers as specified in [Table 4](#). These values tune the L-band tuner to 980 MHz, select a baseband filter with a 10 MHz cutoff frequency, and set the gain to

allow an input signal of approximately -40 dBm to drive the analog-to-digital converters to full scale. You'll probably need to adjust these values for your application.

Maxim 2118		
Register	Value (0x)	Description
0	03	VCO at 4X tuner frequency, $N = 980 \text{ MHz} / 1.25 = 784 = 0x310$ (bits 14–8)
1	10	as above, bits 7–0
2	5D	R counter = div 8 (step size 1.25 MHz), charge pump current = $400\mu\text{A}$, VCO5 (3727 – 4142 MHz)
3	29	$F = 41$ (0x29) for baseband filter 3 dB cutoff of 10 MHz
4	2A	A/D default, high output level, $M=10$ (1 MHz baseband filter clock)
5	0F	normal operation, baseband gain midrange

Table 4. Maxim 2118 Initialization

Writing the Registers

To send a byte to a Maxim 2118 internal register:

1. Clear bit 0 of the [Serial Control Register](#), to indicate a write.
2. Load the desired internal register number (0 through 5) into the [Maxim Address Register](#).
3. Load the desired data byte for the internal register into the Maxim Data Register (see [page 13](#)).
4. Monitor [XFER_BUSY](#) and wait until it's clear before trying the next I²C data transfer.

Reading Status

To read Maxim 2118 status:

1. Set bit 0 of the [Serial Control Register](#), to indicate a read.
2. Write a byte (the value is irrelevant) to the Maxim Data Register (see [page 13](#)) to trigger the I²C read cycle.
3. Monitor [XFER_BUSY](#) and wait until it returns to zero to assure that data transfer is complete.
4. Read the status byte from the Maxim Data Register (see [page 13](#)).

Setting the Baseband Gain

Bits 4–0 of the Maxim 2118 register 5 select one of 32 baseband gain settings. Higher values represent a lower gain. The total adjustment range is approximately 24 dB. (Bits 7–5 of this register are for diagnostic use; for normal operation, clear them all.)

Setting the Baseband Lowpass Filter Cutoff Frequency

Setting the filter bandwidth requires writing to two Maxim 2118 registers — setting the M counter (register four, bits 4–0) such that the crystal frequency divided by M is 1 MHz–2.5 MHz, and then fine-tuning the bandwidth by setting bits 6–0 in Register 3. (Bit 7 in Register 3 is unused; make sure that it's clear.)

With the value of M set to the default 0x0A, the correct value for register 3 is:

$$(\text{cutoff frequency in MHz} - 4) \times 6.9$$

This gives a cutoff frequency range of approximately 4–22 MHz.

For cutoff frequencies above 22 MHz, change the value of M to 5. The correct value for register 3 is then:

$$(\text{cutoff frequency in MHz} - 8) \times 3.45$$

With M = 5, the cutoff frequency can be adjusted from 8–33 MHz. Values that give a cutoff frequency above 33 MHz are not valid.

Setting the Receive Frequency

If you need to change the default value of 1.25 MHz for the phase comparator frequency, write a different R divider value to Maxim register 2. See the Maxim 2118 data sheet for an equation that gives the correct values.

If the default value of 1.25 MHz for the phase comparator frequency is acceptable, setting the receive frequency requires six steps:

1. If the desired frequency is below 1125 MHz, clear bit 7 of the Maxim 2118 register 0, the DIV4 bit, to run the VCO at 4X the receive frequency.

If the desired frequency is 1125 MHz or above, set bit 7 of the Maxim 2118 register 0, the DIV4 bit, to run the VCO at 2X the receive frequency.

2. Program the N divider to the desired receive frequency divided by 1.25:

$$N = (\text{desired receive frequency}) / 1.25$$

The N divider is a 15-bit word: load bits 14–8 into bits 6–0 of Maxim register 0. Load the remaining bits 7–0 of the N divider into bits 7–0 of Maxim register 1.

3. Select the correct VCO for the desired frequency.

The Maxim 2118 has eight VCOs, selected by bits 2–0 of Maxim register 2. The approximate tuning range of each VCO is known, and the manufacturer guarantees that at least one VCO will allow tuning to any given frequency in the range of 915–2175 MHz. However, production tolerances do not permit determining, before the fact, which VCO will work best under a given set of circumstances. Therefore, to find the best VCO for a given frequency requires a process of trial and error.

Select a VCO to start with from [Table 5](#), based on the desired receive frequency. This is guaranteed to be only one off from the final VCO selection.

Table 5. Initial VCO Selection for Receive Frequency

Desired Frequency (GHz)	VCO Number (Maxim Register 2, bits 2–0)	DIV2 bit, bit 7 of Maxim 2118 Register 0
925–931	4	clear
932–1035	5	clear
1036–1123	6	clear
1124–1125	7	clear
1126–1216	0	set
1217–1355	1	set
1356–1512	2	set
1513–1670	3	set
1671–1863	4	set

Table 5. Initial VCO Selection for Receive Frequency (continued)

Desired Frequency (GHz)	VCO Number (Maxim Register 2, bits 2–0)	DIV2 bit, bit 7 of Maxim 2118 Register 0
1864–2071	5	set
2072–2175	6	set

4. Read back the VCO tuning voltage to select a VCO whose tuning voltage is in its midrange when the loop is locked at the desired frequency. To do so:
 - a. Set the Maxim 2118 for the desired receive frequency and the correct initial VCO selection from Table 5 above.
 - b. Wait 100 ms for the VCO and its associated phase-locked loop to stabilize.
 - c. Set the ADC Enable bit ADE (bit 6 of Maxim register 4).
 - d. Wait until `XFER_BUSY` is clear, indicating that the ADC has stabilized.
 - e. Set the ADC Latch bit ADL (bit 7 of Maxim register 4). The tuning voltage value is in bits 4–2 of the status byte.
 - f. Read the Maxim 2118 status as described in [Reading Status on page 10](#).
 - g. Clear the ADC Latch bit ADL (bit 7 of Maxim register 4).
 - h. If the tuning voltage is 111, select the next highest numbered VCO. If the tuning voltage is 000, select the next lowest numbered VCO. Otherwise, the correct VCO is already selected.

NOTE If you change the VCO selection, do not change the setting of the DIV4 bit determined by the desired receive frequency.

- i. If you changed the selected VCO in step h, repeat step b through step g to read the current VCO voltage once more.
5. Set the charge pump gain to the recommended value for tuning voltage that you read with the correct VCO, as shown in Table 6:

Table 6. Charge Pump Gain for Tuning Voltage

Tuning Voltage (bits 4–2, Status)	Charge Pump Gain (bits 4–3, register 2)
001	01
010	01
011	10
100	10
101	11
110	11

6. Clear the ADC Enable bit ADE (bit 6 or Maxim register 4).

Maxim Address Register

Size	8-bit
I/O	read-write
Address	0x55
Access	LBAND_ADDR

Bit	Name	Description
7–0		Address of the Maxim2118 register to which to write the data word in the Maxim Data register.

Maxim Data Register (write)

Size	8-bit
I/O	write only
Address	0x56
Access	LBAND_DATA

Bit	Name	Description
7–0		8-bit data word to write

Maxim Data Register (read)

Size	8-bit
I/O	read only
Address	0x56
Access	LBAND_DATA

Bit	Name	Description
7		0, unused
6	PWR	Set for the first read after powerup.
5		0, unused
4–2	ADC	3-bit value from the VCO tuning voltage analog-to-digital converter inside the Maxim device, used to search for the correct VCO and verify that the VCO loop is locked.
1–0		0, unused

Setting the Gain

Both the IF and the L-band tuners provide variable gain amplifiers at their RF inputs. You set the gain by generating analog voltage levels in a dual-channel DAC:

- The A channel sets the L-band gain.
- The B channel sets the IF gain.

Each DAC channel accepts 12-bit data but ignores the two least significant bits, providing 10-bit resolution. Each DAC output provides a gain control signal adjustable from 0–3 V. Table 7 and Table 8 show the useful gain control voltage ranges for each device. By default, DACA is set to 0x861, providing a channel A gain setting of approximately 1.6 V for the L-band signal, and DACB is set to 0x400, for a channel B gain setting of approximately 0.75 V for the IF signal.

Table 7 provides an approximate mapping of DAC setting to voltage gain from the IF input to the A/D input. A 2 volt pin-to-pin signal at the A/D converter provides full-scale digital output.

Table 7. DAC B Channel Value to IF Signal Gain

Value (0x)	Voltage	IF Gain (db)
0	0	0
199	0.3	15
332	0.6	26
4CB	0.9	38
664	1.2	50
800	1.5	63

Table 8 provides an approximate mapping of DAC setting to voltage gain from the L-band RF input to the A/D input, assuming that the L-band tuner's internal baseband gain register setting is 0x0F — midrange baseband gain.

NOTE Unlike the IF gain, the L-band gain decreases with increased control voltage and DAC value.

Table 8. DAC A Channel Value to L-band Signal Gain

Value (0x)	Voltage	IF Gain (db)
400	0.75	68
555	1.0	66
6A9	1.25	60
800	1.5	50
955	1.75	36
AAA	2.0	24
C00	2.25	10
D52	2.5	–4

You write the gain-control DAC channels through four byte-wide registers: DACA_LOW, DACA_HIGH, DACB_LOW, and DACB_HIGH, described below. After either the DACA_HIGH or DACB_HIGH register is loaded, both DAC channels update with the current contents of their respective registers.

DAC A Low Register

Size	8-bit
I/O	read-write
Address	0x58
Access	DACA_LOW

Bit	Name	Description
7–2	DACA_LOW	Least significant component of voltage specification for the L-band signal gain.
1–0		unused

DAC A High Register

Size	8-bit
I/O	read-write
Address	0x59
Access	DACA_HIGH

Bit	Name	Description
7–4	DACA_HIGH	unused
3–0		Most significant component of voltage specification for the L-band signal gain.

NOTE After either the DAC A High or the DAC B High register is loaded, both A and B channels are updated with the current contents of their respective registers.

DAC B Low Register

Size	8-bit
I/O	read-write
Address	0x5A
Access	DACB_LOW

Bit	Name	Description
7–2	DACA_LOW	Least significant component of voltage specification for the IF signal gain.
1–0		unused

DAC B High Register

Size	8-bit
I/O	read-write
Address	0x5B
Access	DACB_HIGH

Bit	Name	Description
7–4	DACB_HIGH	unused
3–0		Most significant component of voltage specification for the IF signal gain.

NOTE After either the DAC A High or the DAC B High register is loaded, both A and B channels are updated with the current contents of their respective registers.

Acquiring a Signal

You can acquire a signal in DMA mode (normal operation) or test mode, using the Capture Control register.

- In DMA mode, you can read the data acquired.
- In test mode, you can read back the acquired data from the FIFO, one byte at a time, using the Read Data Register.

Capture Control Register (read)

Size	8-bit
I/O	read only
Address	0x63
Access	AD_CAPTURE_CTRL

Bit	Name	Description
7	Capture Busy	When set, A/D data acquisition is in progress, either in test or DMA mode. When clear, no data is being acquired.
6	L-band OTR	Set if the L-band analog-to-digital converter reported an overrange on any data acquired since the last reset by the Fifo Reset bit (2), meaning the input signal exceeded the range of the analog-to-digital converter.
5	IF OTR	Set if the IF analog-to-digital converter reported an overrange on any data acquired since the last reset by the Fifo Reset bit (2), meaning the input signal exceeded the range of the analog-to-digital converter.
4	L-band FIFO overrun	Set if L-band input data FIFO filled during an acquisition and lost data since the last reset of Fifo Reset (bit 2).
3	IF FIFO overrun	Set if IF input data FIFO filled during an acquisition and lost data since the last reset of Fifo Reset (bit 2).
2	FIFO reset	Reads back what was written.

Bit	Name	Description
1	test mode	Set for test mode; clear for DMA mode.
0	Select L-band or IF for test	Set to read back L-band data in test mode; clear to read back IF data in test mode.

Capture Control Register (write)

Size	8-bit
I/O	write only
Address	0x63
Access	AD_CAPTURE_CTRL

Bit	Name	Description
7	Start or stop capture.	Set to start A/D data acquisition in both test and DMA modes. Clear to stop acquisition in DMA mode.
6–3		unused
2	FIFO reset	Toggle (set, then clear) in order to clear the analog-to-digital converter input FIFO before starting signal acquisition. Setting this bit also clears OTR and flag overrun bits 6–3.
1	test mode	Set for test mode; clear for DMA mode.
0	Select L-band or IF for test	Set to read back L-band data in test mode; clear to read back IF data in test mode.

Read Data Register

Size	8-bit
I/O	read-write
Address	0x60
Access	AD_READ_DATA
Comment	Test mode only: Write to advance the FIFO pointers. Read one byte at a time from the IF or L-band FIFO.

Bit	Name	Description
7–0		8-bit data word acquired (DMA mode) or written to read back (test mode)

DMA Mode

To start data transfer:

1. Clear the FIFOs by setting bit 2 of the Capture Control Reg AD_CAPTURE_CTRL , then clearing it. Keep the test mode bit (1) clear throughout this sequence.
2. Set up the PCI SS/GS Xilinx registers for the transfer:
 - For DMA transfer of IF data (DMA channel 0), set CH_EN_REG to 0x01.
 - For DMA transfer of L-band data (DMA channel 1), set CH_EN_REG to 0x02.
 - Or for DMA transfer of both L-band and IF data using both DMA channels 1 and 0, set CH_EN_REG to 0x03.
3. Set the command enable bit (bit 3) by writing a 0x08 in the PCI SS/GS Command Register to enable the DMA interface in the PCI SS/GS Xilinx .
4. Write a 0x80 to the Capture Control Register to start acquiring data in the input FIFOs.
5. Immediately make the appropriate EDT driver call(s) to start DMA on the PCI SS/GS Xilinx.

Test Mode

In test mode, you can capture 2048 32-bit data samples, each of which includes L-band and IF data, to the input data FIFOs. You can read out the test data using programmed IO rather than DMA. Each data sample is four bytes, with the data represented as follows:

bits 31–28	bits 27–16	bits 15–12	bits 11–0
0000	Q channel data	0000	I channel data

The least significant byte of each word reads back first. For each byte, bit 0 of the [Capture Control Register \(write\)](#) selects whether L-band or IF data will be read from the Read Data Register.

To capture and read data in test mode:

1. Clear the FIFO by setting bit 2 of the Capture Control Register AD_CAPTURE_CTRL to 1 and then back to 0.
2. Start the capture by writing an 0x82 to the Capture Control Register to set the test mode and start bits.
3. Monitor the Capture Busy bit (bit 7) in the Capture Control Register until it reads back 0 (not busy).
4. You can read back captured data one byte at a time from the Read Data Register AD_READ_DAT. After each read, advance the internal FIFO pointers to the next byte by writing any value to the Read Data Register.

Both L-band and IF data use the same address pointer so, to read both L-band and IF data, read a byte for each before advancing to the next byte by writing the Read Data Register.

Accessing the Graychips

The SRXL has two GC4016 digital signal processor graychips onboard for digital down-conversion. They are connected to the Xilinx, so your software can access them in whatever manner required.

Each must be programmed separately by writing to 8-bit control registers. To do so:

1. Write the address of the desired control register into the GC_ADDR register.
2. Write the 8-bit data into the GC_DATA register.

NOTE To ensure that data goes to the correct register, be sure to write the address register first — the write to the Graychip occurs immediately after the GC_DATA register has been written.

To read the data from one of the registers:

1. Specify the register to read by writing its address to GC_ADDR.

NOTE The most significant bit — GC_SELECT — determines which Graychip you read.

2. Read the GC_DATA register.

Register descriptions for the [GC4016](#)

The sample application allows you to run a diagnostic checksum on the Graychips.

Graychip Address Register

Size	8-bit
I/O	read-write
Address	0x64
Access	GC_ADDR

Bit	Name	Description
7	GC_SELECT	When clear, selects Graychip number 1. When set, selects Graychip number 2.
6–5		not used
4–0	A[4–0]	Address of register to read from or write to. Most significant bit is bit 4; least significant is bit 0.

Graychip Data Register

Size	8-bit
I/O	read-write
Address	0x65
Access	GC_DATA

Bit	Name	Description
7–0	GC_DATA	Contains the data to write to the register specified in the Graychip Address register, or the data read from it.

Pinouts

The Satellite Tuner connects to the main board using five 64-pin connectors, P1 – P5, whose locations are shown in Figure 1.

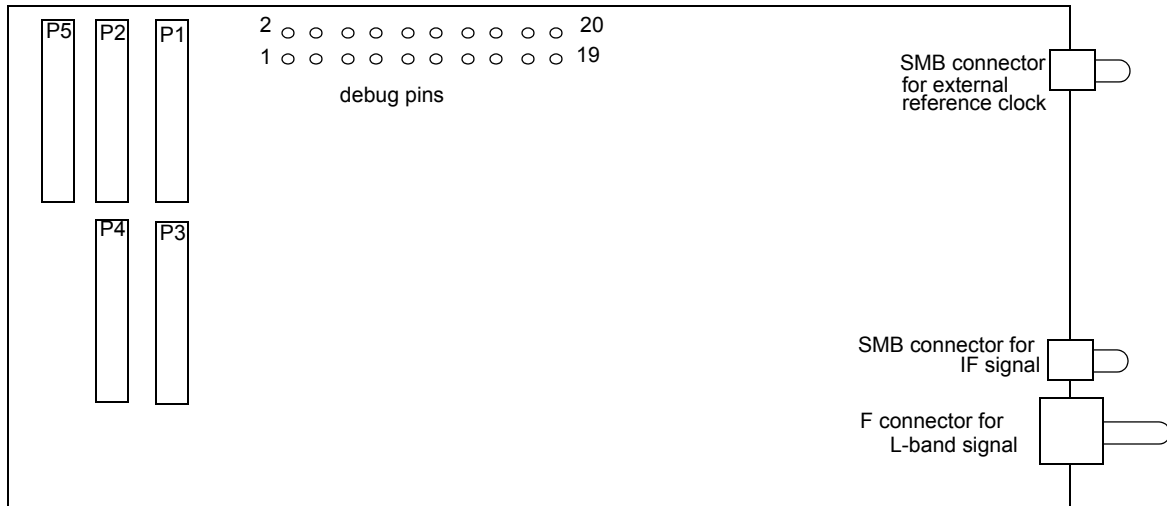


Figure 1. Connector Locations

Table 9 and Table 10 show the signals from the SRXL Xilinx to these connectors, and from there to the user interface Xilinx on the PCI SS/GS main board.

Table 11 shows the local signals from the SRXL Xilinx to the Graychips.

Table 12 shows the local signals from the SRXL Xilinx to the other devices on the SRXL mezzanine board.

Legend

free A wire connects the SRXL Xilinx to the user interface Xilinx on the main board. Your firmware can access these signals.

unused No wire connects the SRXL Xilinx to any outside device. These signals are inaccessible.

BD IDx A board identification signal, for use with the PCI SS/GS Board ID register.

SRXL signal names in Table 9 and Table 10 are identical to those used in the VHDL bitfile `srxl_top.bit`.

NOTE The two external board ID bits marked with an asterisk * are not in the bitfile.

Table 9. SRXL to PCI GS Connector Pinout

Connector Pin	PCI GS Pin	SRXL Xilinx Pin	SRXL Signal	Connector Pin	PCI GS Pin	SRXL Xilinx Pin	SRXL Signal
P1.1	E30	D3	Xilinx PROG_B	P1.46	P38	AE11	pad_ch0_dataout[24]
P1.2			unused	P1.47	M38	AE12	pad_ch0_dataout[25]
P1.3			ground	P1.48	R38	AF11	pad_ch0_dataout[26]
P1.4	F30	W2	pad_ch0_outval_l	P1.49	R39	AF12	pad_ch0_dataout[27]
P1.5	C29	W1	pad_ch0_dataout[0]	P1.50			+5 V
P1.6	F29	Y2	pad_ch0_dataout[1]	P1.51			ground
P1.7	G30		BD ID0	P1.52	T37	AF13	pad_ch0_dataout[28]
P1.8			+5 V	P1.53	T38	AF15	pad_ch0_dataout[29]
P1.9	C33	AA2	pad_ch0_dataout[2]	P1.54	U38	AE15	pad_ch0_dataout[30]
P1.10	D31	Y1	pad_ch0_dataout[3]	P1.55	U39	AE16	pad_ch0_dataout[31]
P1.11			ground	P1.56			ground
P1.12	E31	AA1	pad_ch0_dataout[4]	P1.57			unused
P1.13	C35	AB2	pad_ch0_dataout[5]	P1.58	V38	AF16	pad_ch0_out64_l
P1.14			ground	P1.59	V39	AF17	pad_ch1_out64_l
P1.15			ground	P1.60	Y36	AE17	pad_ch1_outval_l
P1.16	D35	AB1	pad_ch0_dataout[6]	P1.61	Y37	AE18	pad_ch1_en_out_l
P1.17	D34	AC2	pad_ch0_dataout[7]	P1.62			+5 V
P1.18			+5 V	P1.63			ground
P1.19			unused	P1.64	D20	AD14	pci_clk_out
P1.20	E34	AC1	pad_ch0_dataout[8]	P2.1			unused
P1.21	D33	AD1	pad_ch0_dataout[9]	P2.2	D28	W3	free
P1.22	F33	AD2	pad_ch0_dataout[10]	P2.3	E28	Y4	free
P1.23	G31	AE4	pad_ch0_dataout[11]	P2.4	G29	W4	free
P1.24			ground	P2.5	H29	AA3	free
P1.25			ground	P2.6			ground
P1.26	H31	AF4	pad_ch0_dataout[12]	P2.7			ground
P1.27	H38	AF5	pad_ch0_dataout[13]	P2.8	H33	AA4	free
P1.28	H37	AE5	pad_ch0_dataout[14]	P2.9	H34	AB4	free
P1.29	J39	AE6	pad_ch0_dataout[15]	P2.10	J33	AB3	free
P1.30			+5 V	P2.11	H30		BD ID1
P1.31			unused	P2.12			unused
P1.32	J38	AF6	pad_ch0_dataout[16]	P2.13	J34	AD4	free
P1.33	K39	AE7	pad_ch0_dataout[17]	P2.14	J31		BD ID2
P1.34			ground	P2.15			unused
P1.35			ground	P2.16	H32		BD ID3
P1.36	K38	AF7	pad_ch0_dataout[18]	P2.17	H36	AC5	free
P1.37	L39	AE8	pad_ch0_dataout[19]	P2.18			ground
P1.38			+5 V	P2.19	K36	AC6	free
P1.39			ground	P2.20	K35	AD5	free
P1.40	L38	AF8	pad_ch0_dataout[20]	P2.21			ground
P1.41	N39	AE10	pad_ch0_dataout[21]	P2.22	J35	AD6	free
P1.42	N38	AE9	pad_ch0_dataout[22]	P2.23	L34	AB7	free
P1.43	P39	AF10	pad_ch0_dataout[23]	P2.24			unused
P1.44			ground	P2.25	L35	AD8	free
P1.45			unused	P2.26	J37	AC7	free

Table 9. SRXL to PCI GS Connector Pinout (continued)

Connector Pin	PCI GS Pin	SRXL Xilinx Pin	SRXL Signal	Connector Pin	PCI GS Pin	SRXL Xilinx Pin	SRXL Signal
P2.27			ground	P3.8			ground
P2.28	K37	AC8	free	P3.9			unused
P2.29	L37	AD9	free	P3.10	AB38	AE23	pad_ch1_dataout[26]
P2.30			ground	P3.11	AB37	AE24	pad_ch1_dataout[25]
P2.31	L36	AD10	free	P3.12	AB36	AF24	pad_ch1_dataout[24]
P2.32	M34	AC9	free	P3.13	AC39	AD25	pad_ch1_dataout[23]
P2.33			ground	P3.14			ground
P2.34	M33	AC10	free	P3.15			ground
P2.35	P36	AC11	free	P3.16	AC38	AD26	Xilinx CCLK
P2.36			unused	P3.17	AC36	AC25	pad_ch1_dataout[21]
P2.37			ground	P3.18	AD36	AC26	pad_ch1_dataout[20]
P2.38	P35	AD12	free	P3.19	AD38	AB25	pad_ch1_dataout[19]
P2.39	P37	AB12	free	P3.20			ground
P2.40			ground	P3.21			+3.3 V
P2.41			unused	P3.22	AD37	AB26	pad_ch1_dataout[18]
P2.42	N37	AC13	free	P3.23	AE39	AA25	pad_ch1_dataout[17]
P2.43	R37	AB13	free	P3.24	AE38	AA26	pad_ch1_dataout[16]
P2.44			ground	P3.25	AH38	Y25	pad_ch1_dataout[15]
P2.45	U36	AC14	Xilinx INIT	P3.26			ground
P2.46	R36	AF14	pci_clk_in	P3.27			ground
P2.47			ground	P3.28	AF38	Y26	pad_ch1_dataout[14]
P2.48	T36	AD15	free	P3.29	AF37	Y21	pad_ch1_dataout[13]
P2.49	Y33	AC16	free	P3.30	AG37	AF23	pad_ch1_dataout[12]
P2.50			unused	P3.31	AG39	W25	pad_ch1_dataout[11]
P2.51	W35	AC17	free	P3.32			ground
P2.52	V36	AB16	free	P3.33			ground
P2.53			ground	P3.34	AG38	AF20	pad_ch1_dataout[10]
P2.54	W34	AD17	free	P3.35	AG35	W26	pad_ch1_dataout[9]
P2.55	W37	W16	free	P3.36	AH36	AF19	pad_ch1_dataout[8]
P2.56			ground	P3.37	AF39	V25	pad_ch1_dataout[7]
P2.57	W31	Y16	free	P3.38			ground
P2.58	W36	AA16	free	P3.39			unused
P2.59			ground	P3.40	AK38	AE19	pad_ch1_dataout[6]
P2.60	Y31	Y17	free	P3.41	AJ39	U26	pad_ch1_dataout[5]
P2.61	V37	AA17	free	P3.42	AJ38	V20	pad_ch1_dataout[4]
P2.62			unused	P3.43	AJ37	U25	pad_ch1_dataout[3]
P2.63			ground	P3.44			ground
P2.64	Y32	AB17	free	P3.45			ground
P3.1	AA37	AE20	pad_ch1_dataout[31]	P3.46	AJ36	W23	pad_ch1_dataout[2]
P3.2			ground	P3.47	AJ35	T26	pad_ch1_dataout[1]
P3.3			ground	P3.48	AJ34	V24	pad_ch1_dataout[0]
P3.4	AA36	AE21	pad_ch1_dataout[30]	P3.49	AK39	T25	pad_ch1_dataout[22]
P3.5	AA35	AF21	pad_ch1_dataout[29]	P3.50			ground
P3.6	AA34	AE22	pad_ch1_dataout[28]	P3.51			ground
P3.7	AB39	AF22	pad_ch1_dataout[27]	P3.52	AH37	U23	pad_ch0_en_out_l

Table 9. SRXL to PCI GS Connector Pinout (continued)

Connector Pin	PCI GS Pin	SRXL Xilinx Pin	SRXL Signal	Connector Pin	PCI GS Pin	SRXL Xilinx Pin	SRXL Signal
P3.53	AK36	R26	ind_io[7]	P4.34	AJ31	U21	free
P3.54	AK35	R24	ind_io[6]	P4.35	AJ33	T21	free
P3.55	AK37	R25	ind_io[5]	P4.36			ground
P3.56			ground	P4.37	AK34	AC24	Xilinx DONE
P3.57			unused	P4.38	AJ32	R21	free
P3.58	AL37	AB20	ind_io[4]	P4.39	AK32	AB24	free
P3.59	AL39	P26	ind_io[3]	P4.40	AK33	W20	free
P3.60	AL38	P23	ind_io[2]	P4.41			ground
P3.61	AM38	P25	ind_io[1]	P4.42	AK31	U20	free
P3.62			ground	P4.43	AL34	AB23	free
P3.63			ground	P4.44	AL33	T20	free
P3.64	AM37	Y20	ind_io[0]	P4.45	AL35	AA24	free
P4.1			unused	P4.46			ground
P4.2	Y29	AC18	free	P4.47	AM33	AA23	free
P4.3	AA31	AB18	free	P4.48	AM36	R20	free
P4.4	AA30	AD18	free	P4.49	AN34	Y23	free
P4.5	AA33	AA18	free	P4.50	AM34	W24	free
P4.6			ground	P4.51			ground
P4.7	AB33	Y18	free	P4.52	AN33	U24	free
P4.8	AA32	AC19	free	P4.53	AL31	T23	free
P4.9	AC31	AD19	free	P4.54	AM32	R22	free
P4.10	AB32	AB19	free	P4.55	AH29	P24	free
P4.11			ground	P4.56			ground
P4.12	AC30	AA19	free	P4.57	AP33	AC20	free
P4.13	AB35	Y19	free	P4.58	AN31	AA20	free
P4.14	AB34	AD23	free	P4.59			unused
P4.15	AC35	AD22	free	P4.60			unused
P4.16			ground	P4.61			ground
P4.17	AD34	AB22	free	P4.62			unused
P4.18	AC34	AC22	free	P4.63			unused
P4.19	AE35	Y22	free	P4.64			unused
P4.20	AD33	AA22	free	P5.1			unused
P4.21			ground	P5.2	C21	W5	ind_ctl[2]
P4.22	AE34	W22	free	P5.3	C22	Y5	ind_ctl[1]
P4.23	AF36	AD21	free	P5.4	E21	W6	ind_ctl[0]
P4.24	AF35	AC21	free	P5.5	D21	Y6	free
P4.25	AG34	V22	free	P5.6			ground
P4.26			ground	P5.7	F21	AB5	free
P4.27	AG31	U22	free	P5.8	F20	Y7	free
P4.28	AG33	AB21	free	P5.9	G22	AA6	free
P4.29	AH34	T22	free	P5.10	G21	AB6	free
P4.30	AH32	AA21	free	P5.11			ground
P4.31			ground	P5.12	E22	AA7	free
P4.32	AH33	W21	free	P5.13	D22	AA8	free
P4.33	AH30	V21	free	P5.14	F22	AB8	free

Table 9. SRXL to PCI GS Connector Pinout (continued)

Connector Pin	PCI GS Pin	SRXL Xilinx Pin	SRXL Signal	Connector Pin	PCI GS Pin	SRXL Xilinx Pin	SRXL Signal
P5.15	E23	Y8	free	P5.40	K34		extbdid CLK *
P5.16			ground	P5.41			ground
P5.17	C23	AA9	free	P5.42	L32		extbdid DATA *
P5.18	D23	AB9	free	P5.43	M32	W13	free
P5.19	F24	AA10	free	P5.44			unused
P5.20	G23	Y9	free	P5.45	N34	AB14	free
P5.21			ground	P5.46			ground
P5.22	E24	AB10	free	P5.47	P34	AA14	free
P5.23	E25	AB11	free	P5.48			unused
P5.24	J24	Y10	free	P5.49	N20	Y14	free
P5.25	H24	AA11	free	P5.50			unused
P5.26			ground	P5.51			ground
P5.27	C25	W11	free	P5.52			unused
P5.28	D25	Y11	free	P5.53	R35	W14	free
P5.29	E26	AA12	free	P5.54			unused
P5.30			unused	P5.55	T34	AB15	free
P5.31			ground	P5.56			ground
P5.32			unused	P5.57	U35	AA15	free
P5.33	F25	Y12	free	P5.58			unused
P5.34			unused	P5.59	V35	Y15	Xilinx DIN
P5.35	C26	W12	free	P5.60			unused
P5.36			ground	P5.61			ground
P5.37	K33	AA13	free	P5.62			unused
P5.38			unused	P5.63	U33	W15	free
P5.39	L33	Y13	free	P5.64			unused

Table 10. SRXL to PCI SS Connector Pinout

Connector Pin	PCI SS Pin	SRXL Xilinx Pin	SRXL Signal	Connector Pin	PCI SS Pin	SRXL Xilinx Pin	SRXL Signal
P1.1	N36	D3	Xilinx PROG_B	P1.46	AW26	AE11	pad_ch0_dataout[24]
P1.2			unused	P1.47	AW25	AE12	pad_ch0_dataout[25]
P1.3			ground	P1.48	AW24	AF11	pad_ch0_dataout[26]
P1.4	N37	W2	pad_ch0_outval_l	P1.49	AW23	AF12	pad_ch0_dataout[27]
P1.5	P36	W1	pad_ch0_dataout[0]	P1.50			+5 V
P1.6	P38	Y2	pad_ch0_dataout[1]	P1.51			ground
P1.7	AU29		BD ID0	P1.52	AW22	AF13	pad_ch0_dataout[28]
P1.8			+5 V	P1.53	AW21	AF15	pad_ch0_dataout[29]
P1.9	P39	AA2	pad_ch0_dataout[2]	P1.54	AU28	AE15	pad_ch0_dataout[30]
P1.10	R36	Y1	pad_ch0_dataout[3]	P1.55	AU27	AE16	pad_ch0_dataout[31]
P1.11			ground	P1.56			ground
P1.12	R38	AA1	pad_ch0_dataout[4]	P1.57			unused
P1.13	R39	AB2	pad_ch0_dataout[5]	P1.58	AU26	AF16	pad_ch0_out64_l
P1.14			ground	P1.59	AU25	AF17	pad_ch1_out64_l
P1.15			ground	P1.60	AU24	AE17	pad_ch1_outval_l
P1.16	T36	AB1	pad_ch0_dataout[6]	P1.61	AU23	AE18	pad_ch1_en_out_l
P1.17	T37	AC2	pad_ch0_dataout[7]	P1.62			+5 V
P1.18			+5 V	P1.63			ground
P1.19			unused	P1.64	AU22	AD14	pci_clk_out
P1.20	T38	AC1	pad_ch0_dataout[8]	P2.1			unused
P1.21	T39	AD1	pad_ch0_dataout[9]	P2.2	U39	W3	free
P1.22	U35	AD2	pad_ch0_dataout[10]	P2.3	V35	Y4	free
P1.23	U36	AE4	pad_ch0_dataout[11]	P2.4	V36	W4	free
P1.24			ground	P2.5	V37	AA3	free
P1.25			ground	P2.6			ground
P1.26	U37	AF4	pad_ch0_dataout[12]	P2.7			ground
P1.27	U38	AF5	pad_ch0_dataout[13]	P2.8	V38	AA4	free
P1.28	AW36	AE5	pad_ch0_dataout[14]	P2.9	V39	AB4	free
P1.29	AW35	AE6	pad_ch0_dataout[15]	P2.10	W36	AB3	free
P1.30			+5 V	P2.11	AU30		BD ID1
P1.31			unused	P2.12			unused
P1.32	AW34	AF6	pad_ch0_dataout[16]	P2.13	W37	AD4	free
P1.33	AW33	AE7	pad_ch0_dataout[17]	P2.14	AT30		BD ID2
P1.34			ground	P2.15			unused
P1.35			ground	P2.16	AT34		BD ID3
P1.36	AW32	AF7	pad_ch0_dataout[18]	P2.17	W38	AC5	free
P1.37	AW31	AE8	pad_ch0_dataout[19]	P2.18			ground
P1.38			+5 V	P2.19	W39	AC6	free
P1.39			ground	P2.20	Y38	AD5	free
P1.40	AW30	AF8	pad_ch0_dataout[20]	P2.21			ground
P1.41	AW29	AE10	pad_ch0_dataout[21]	P2.22	AA36	AD6	free
P1.42	AW28	AE9	pad_ch0_dataout[22]	P2.23	AA37	AB7	free
P1.43	AW27	AF10	pad_ch0_dataout[23]	P2.24			unused
P1.44			ground	P2.25	AA38	AD8	free
P1.45			unused	P2.26	AA39	AC7	free

Table 10. SRXL to PCI SS Connector Pinout (continued)

Connector Pin	PCI SS Pin	SRXL Xilinx Pin	SRXL Signal	Connector Pin	PCI SS Pin	SRXL Xilinx Pin	SRXL Signal
P2.27			ground	P3.8			ground
P2.28	AB37	AC8	free	P3.9			unused
P2.29	AV35	AD9	free	P3.10	AV17	AE23	pad_ch1_dataout[26]
P2.30			ground	P3.11	AW15	AE24	pad_ch1_dataout[25]
P2.31	AU34	AD10	free	P3.12	AV16	AF24	pad_ch1_dataout[24]
P2.32	AV34	AC9	free	P3.13	AW14	AD25	pad_ch1_dataout[23]
P2.33			ground	P3.14			ground
P2.34	AV33	AC10	free	P3.15			ground
P2.35	AV32	AC11	free	P3.16	AV15	AD26	Xilinx CCLK
P2.36			unused	P3.17	AW13	AC25	pad_ch1_dataout[21]
P2.37			ground	P3.18	AV14	AC26	pad_ch1_dataout[20]
P2.38	AV31	AD12	free	P3.19	AW12	AB25	pad_ch1_dataout[19]
P2.39	AV30	AB12	free	P3.20			ground
P2.40			ground	P3.21			+3.3 V
P2.41			unused	P3.22	AV13	AB26	pad_ch1_dataout[18]
P2.42	AV29	AC13	free	P3.23	AW11	AA25	pad_ch1_dataout[17]
P2.43	AV28	AB13	free	P3.24	AV12	AA26	pad_ch1_dataout[16]
P2.44			ground	P3.25	AW10	Y25	pad_ch1_dataout[15]
P2.45	AV27	AC14	Xilinx INIT	P3.26			ground
P2.46	AV26	AF14	pci_clk_in	P3.27			ground
P2.47			ground	P3.28	AV11	Y26	pad_ch1_dataout[14]
P2.48	AV25	AD15	free	P3.29	AW9	Y21	pad_ch1_dataout[13]
P2.49	AV24	AC16	free	P3.30	AV10	AF23	pad_ch1_dataout[12]
P2.50			unused	P3.31	AW8	W25	pad_ch1_dataout[11]
P2.51	AV23	AC17	free	P3.32			ground
P2.52	AV22	AB16	free	P3.33			ground
P2.53			ground	P3.34	AV9	AF20	pad_ch1_dataout[10]
P2.54	AV21	AD17	free	P3.35	AW7	W26	pad_ch1_dataout[9]
P2.55	AT27	W16	free	P3.36	AV8	AF19	pad_ch1_dataout[8]
P2.56			ground	P3.37	AW6	V25	pad_ch1_dataout[7]
P2.57	AT26	Y16	free	P3.38			ground
P2.58	AT25	AA16	free	P3.39			unused
P2.59			ground	P3.40	AV7	AE19	pad_ch1_dataout[6]
P2.60	AT24	Y17	free	P3.41	AW5	U26	pad_ch1_dataout[5]
P2.61	AT23	AA17	free	P3.42	AV6	V20	pad_ch1_dataout[4]
P2.62			unused	P3.43	AW4	U25	pad_ch1_dataout[3]
P2.63			ground	P3.44			ground
P2.64	AT22	AB17	free	P3.45			ground
P3.1	AW18	AE20	pad_ch1_dataout[31]	P3.46	AV5	W23	pad_ch1_dataout[2]
P3.2			ground	P3.47	AV4	T26	pad_ch1_dataout[1]
P3.3			ground	P3.48	AV3	V24	pad_ch1_dataout[0]
P3.4	AV19	AE21	pad_ch1_dataout[30]	P3.49	AU4	T25	pad_ch1_dataout[22]
P3.5	AW17	AF21	pad_ch1_dataout[29]	P3.50			ground
P3.6	AV18	AE22	pad_ch1_dataout[28]	P3.51			ground
P3.7	AW16	AF22	pad_ch1_dataout[27]	P3.52	AR4	U23	pad_ch0_en_out_l

Table 10. SRXL to PCI SS Connector Pinout (continued)

Connector Pin	PCI SS Pin	SRXL Xilinx Pin	SRXL Signal	Connector Pin	PCI SS Pin	SRXL Xilinx Pin	SRXL Signal
P3.53	AT3	R26	ind_io[7]	P4.34	AU8	U21	free
P3.54	AR3	R24	ind_io[6]	P4.35	AT8	T21	free
P3.55	AT2	R25	ind_io[5]	P4.36			ground
P3.56			ground	P4.37	AU7	AC24	Xilinx DONE
P3.57			unused	P4.38	AT7	R21	free
P3.58	AR2	AB20	ind_io[4]	P4.39	AU6	AB24	free
P3.59	AT1	P26	ind_io[3]	P4.40	AT6	W20	free
P3.60	AK1	P23	ind_io[2]	P4.41			ground
P3.61	AR1	P25	ind_io[1]	P4.42	AP4	U20	free
P3.62			ground	P4.43	AP3	AB23	free
P3.63			ground	P4.44	AP2	T20	free
P3.64	AJ4	Y20	ind_io[0]	P4.45	AP1	AA24	free
P4.1			unused	P4.46			ground
P4.2	AW20	AC18	free	P4.47	AN4	AA23	free
P4.3	AU21	AB18	free	P4.48	AN3	R20	free
P4.4	AT21	AD18	free	P4.49	AN2	Y23	free
P4.5	AU19	AA18	free	P4.50	AN1	W24	free
P4.6			ground	P4.51			ground
P4.7	AT19	Y18	free	P4.52	AM4	U24	free
P4.8	AU18	AC19	free	P4.53	AM3	T23	free
P4.9	AT18	AD19	free	P4.54	AM2	R22	free
P4.10	AR18	AB19	free	P4.55	AM1	P24	free
P4.11			ground	P4.56			ground
P4.12	AU17	AA19	free	P4.57	AL4	AC20	free
P4.13	AT17	Y19	free	P4.58	AL3	AA20	free
P4.14	AR17	AD23	free	P4.59			unused
P4.15	AU16	AD22	free	P4.60			unused
P4.16			ground	P4.61			ground
P4.17	AT16	AB22	free	P4.62			unused
P4.18	AU15	AC22	free	P4.63			unused
P4.19	AT15	Y22	free	P4.64			unused
P4.20	AU14	AA22	free	P5.1			unused
P4.21			ground	P5.2	AB38	W5	ind_ctl[2]
P4.22	AT14	W22	free	P5.3	AC36	Y5	ind_ctl[1]
P4.23	AU13	AD21	free	P5.4	AC37	W6	ind_ctl[0]
P4.24	AT13	AC21	free	P5.5	AC38	Y6	free
P4.25	AU12	V22	free	P5.6			ground
P4.26			ground	P5.7	AC39	AB5	free
P4.27	AU11	U22	free	P5.8	AD36	Y7	free
P4.28	AT11	AB21	free	P5.9	AD37	AA6	free
P4.29	AU10	T22	free	P5.10	AD38	AB6	free
P4.30	AT10	AA21	free	P5.11			ground
P4.31			ground	P5.12	AD39	AA7	free
P4.32	AU9	W21	free	P5.13	AE37	AA8	free
P4.33	AT9	V21	free	P5.14	AE38	AB8	free

Table 10. SRXL to PCI SS Connector Pinout (continued)

Connector Pin	PCI SS Pin	SRXL Xilinx Pin	SRXL Signal	Connector Pin	PCI SS Pin	SRXL Xilinx Pin	SRXL Signal
P5.15	AF36	Y8	free	P5.40	AM38		extbdid CLK *
P5.16			ground	P5.41			ground
P5.17	AF37	AA9	free	P5.42	AN36		extbdid DATA *
P5.18	AF38	AB9	free	P5.43	AN37	W13	free
P5.19	AF39	AA10	free	P5.44			unused
P5.20	AG36	Y9	free	P5.45	AN39	AB14	free
P5.21			ground	P5.46			ground
P5.22	AG37	AB10	free	P5.47	AP36	AA14	free
P5.23	AG39	AB11	free	P5.48			unused
P5.24	AH37	Y10	free	P5.49	AP38	Y14	free
P5.25	AH38	AA11	free	P5.50			unused
P5.26			ground	P5.51			ground
P5.27	AJ36	W11	free	P5.52			unused
P5.28	AJ37	Y11	free	P5.53	AR37	W14	free
P5.29	AJ38	AA12	free	P5.54			unused
P5.30			unused	P5.55	AR39	AB15	free
P5.31			ground	P5.56			ground
P5.32			unused	P5.57	AT38	AA15	free
P5.33	AK38	Y12	free	P5.58			unused
P5.34			unused	P5.59	AU36	Y15	Xilinx DIN
P5.35	AL36	W12	free	P5.60			unused
P5.36			ground	P5.61			ground
P5.37	AL38	AA13	free	P5.62			unused
P5.38			unused	P5.63	AR23	W15	free
P5.39	AM37	Y13	free	P5.64			unused

Table 11. SRXL Xilinx to Graychip Pinout

Graychip Pin Name	Graychip Pin Number	SRXL Xilinx Pin to Graychip 1	SRXL Xilinx Pin to Graychip 2	Graychip Pin Name	Graychip Pin Number	SRXL Xilinx Pin to Graychip 1	SRXL Xilinx Pin to Graychip 2
AIN[0]	D14	B10	G22	DIN[10]	F1	K1	A15
AIN[1]	E13	P4	F23	DIN[11]	F2	K2	F15
AIN[2]	E12	P3	H26	DIN[12]	E3	L7	F14
AIN[3]	E14	A10	G23	DIN[13]	E1	L2	B15
AIN[4]	F13	E9	G26	C[0]	B9	P6	J20
AIN[5]	F12	C9	E23	C[1]	A9	N1	J21
AIN[6]	F14	B9	H25	C[2]	C9	P7	H21
AIN[7]	G13	F8	F26	C[3]	B8	N2	H12
AIN[8]	G14	G8	G25	C[4]	C8	N5	E11
AIN[9]	H12	G6	E21	C[5]	A7	N4	A11
AIN[10]	H11	F6	D20	C[6]	B7	N3	E12
AIN[11]	H14	G7	F25	C[7]	A6	M1	B11
AIN[12]	H13	D7	E26	P[0]	B11	R7	H23
AIN[13]	J12	E6	D21	P[1]	C11	R5	H22
BIN[0]	J14	C8	E25	P[2]	A11	R6	J23

