



[Open this model](#)

MIMO FlexOFDM

The MIMO FlexOFDM example implements a unidirectional MIMO wireless transmission at 2.4 GHz (or 5 GHz) using the orthogonal frequency-division multiplexing (OFDM) scheme. The transmitter sends a PN15 sequence and the received QPSK pattern is viewed with Simulink, through FPGA HIL co-simulation.

Important

This example is based on the MIMO FlexOFDM core (PHY layer) designed by a special engineering group at Xilinx. The example is fully capable of real-time implementation and simulation. Lyrtech does not support problems caused by the FlexOFDM core or its implementation, as Lyrtech does not own this IP. Lyrtech only supports the proper functioning of the example when it is used “as is” (i.e. without modifications to the default configuration parameters). All source files are supplied with the example, accompanied with documentation for the example’s setup. Users are encouraged to explore the supplied files to expand their knowledge of the core implementation. The MIMO FlexOFDM example is only supported by Lyrtech VHS-ADC/VHS-DAC equipped with SX55 Virtex-4 FPGA.

Introduction

FlexOFDM is a multi-antenna communication system with configurable parameters, such as the number of subcarriers, MIMO schemes such as:

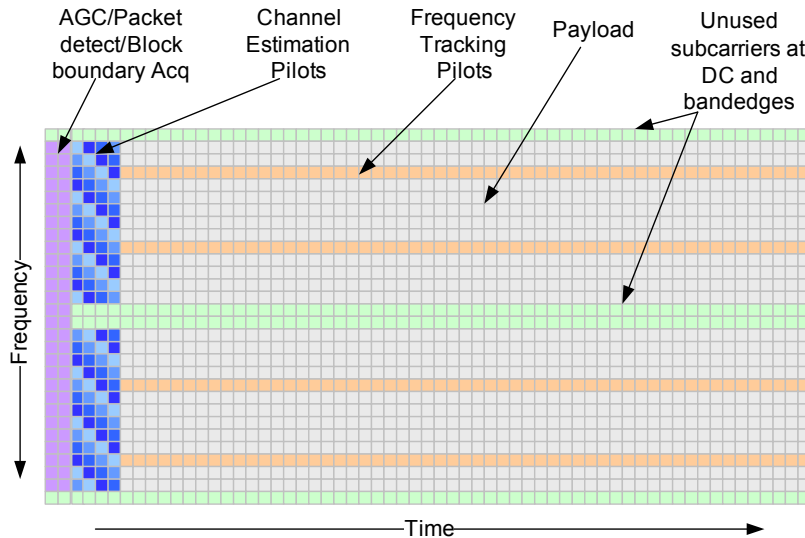
- * orthogonal/semi-orthogonal STBC;
- * spatial multiplexing;
- * interference cancellation STBC;

to provide a range of tradeoffs between throughput and reliability. The goal of FlexOFDM is to provide an “at-scale” example of a wireless communication system on a platform FPGA, using DSP48s for the PHY and signal processing operations. It is a great stepping stone for developing MIMO–OFDM communication systems on FPGAs.

Description of the communication system

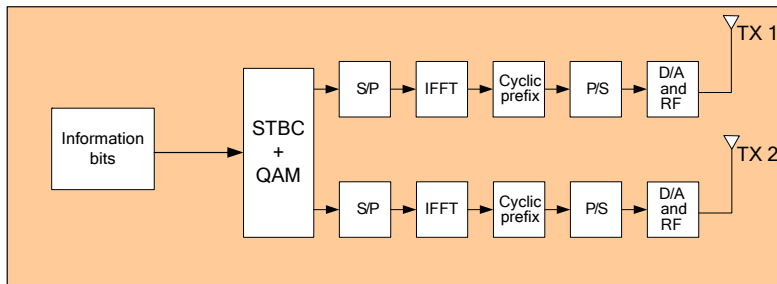
The goal of this communication system is to offer a highly configurable communication system capable of operating in a variety of environments. The size of FFTs can vary from 64 subcarriers to 2048 subcarriers (64, 128, 256, 512, 1024, and 2048; default: 64). The model also allows you to select 4, 16, or 64 QAMs (default: 4). The actual system supports 2×2 antennas (Alamouti system). The packet structure is also programmable. The system initially assumes a quasi-static channel (*i.e.* the channel is assumed to be constant during the duration of the packet—somewhat like in 802.11a system). With the necessary pilot modifications for WiMAX, however, the system can be turned into one supporting high vehicular speeds.

The system operates in packet mode with a preamble of four symbols (for a 2x2 system). The packet structure appears in the figure below. The two first, 32- μ s OFDM symbols are dedicated to AGC, packet detection, block boundary detection, and coarse carrier frequency offset estimation. The following two OFDM symbols are used in 2x2 MIMO channel matrix estimation. Continuous pilots are embedded in specific subcarriers (up to a maximum of 16) for carrier frequency offset tracking and sampler frequency offset tracking, as well as to estimate and correct the common phase error portion of phase noise. There are unused subcarriers at DC and at the band edges allowing for non-ideal filters in the analog front end. The overall packet length is maintained shorter than the coherence time of the channel.



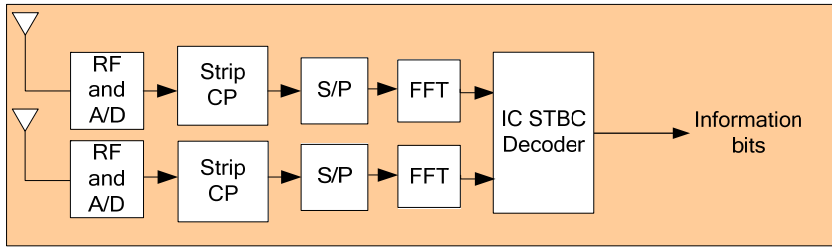
Packet structure

The block diagrams of the communication system appear below. The information bits are sent into a space-time block coding module, and the data stream is OFDM-encoded and transmitted over two antennas. **The initial version of the system is an uncoded (no FEC) communication system.**



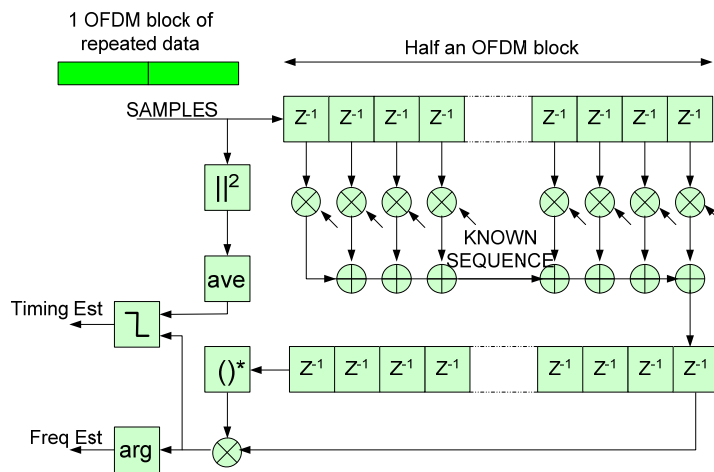
Transmitter block diagram

At the receiver, the two signals received are down-converted to baseband, and OFDM demodulated. An interference cancellation STBC decodes the data stream. The block diagram of the receiver appears here.



Receiver block diagram

The first part of the receiver is a packet detection module and a block boundary detection module. The block boundary detection module is a matched filter to the training symbols, which is illustrated in the figure below.



Block boundary detection algorithm

The first part of the block boundary detection module is a correlator that is half the length of the OFDM symbol ($N_s/2$). The correlation is performed between the received samples and the known PN sequence in the time domain. When the received samples coincide with the known sequence, a large peak occurs. Later, the conjugation of the correlator output is multiplied by the correlator output $N_s/2$ samples. This generates a timing metric signal that displays a sharp peak with a phase proportional to the carrier frequency offset. This peak is compared to the average received power—if the threshold is exceeded, the block boundary is detected and the frequency estimate is available in the phase of the timing metric. This operation also requires a preamble of the same structure as the packet detection algorithm with a two-symbol header of the same preamble structure to aid packet detection and block boundary detection.

Essential parts of the receiver are carrier frequency acquisition and tracking. A coarse estimate of the CFO is obtained in the block boundary detection module. The post-FFT-based carrier frequency offset estimator requires pilots to estimate the CFO. In the presence of the CFO, each subcarrier of an OFDM block is rotated by the same amount. The amount of angular rotation from one OFDM block to the next is proportional to the CFO. Continuous pilots are used to compute the angular rotation of each subcarrier.

The choice of the IC-STBC scheme emphasizes a three-way tradeoff between throughput, reliability, and implementation complexity. The IC-STBC scheme decodes two STBC-encoded

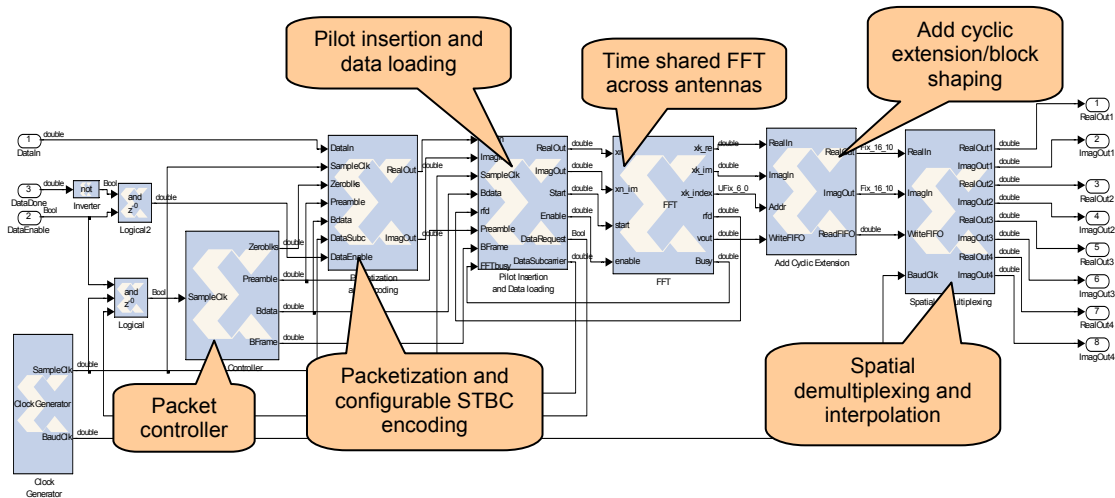
data streams. The MMSE implementation of the IC-STBC decoder is formulated below for a 4x2 case.

Note

Because there is only one interfering data stream for this scheme, two receive antennas are enough to cancel the interferer. With four receiver antennas, additional receive diversity is available, yielding higher performance.

The CFO estimation is available at the block boundary detection phase. The fine carrier frequency offset estimation is performed through the help of continuous pilots.

FlexOFDM transmitter



System Generator for DSP FlexOFDM transmitter model

The System Generator for DSP FlexOFDM transmitter implementation appears. The same OFDM modulator is time-multiplexed for all four transmission antennas. The transmitter must operate at a system clock rate-to-bandwidth ratio of 1:4 for a 2x2 system and 1:8 for a 4x4 system.

Transmitter resource usage

FPGA	Input precision	FFT precision	FFT size	TX number	DSP48/ 18x18 mults	Slices	BRAMs
Virtex-4	<16,10>	12	64	2	25	2594	26

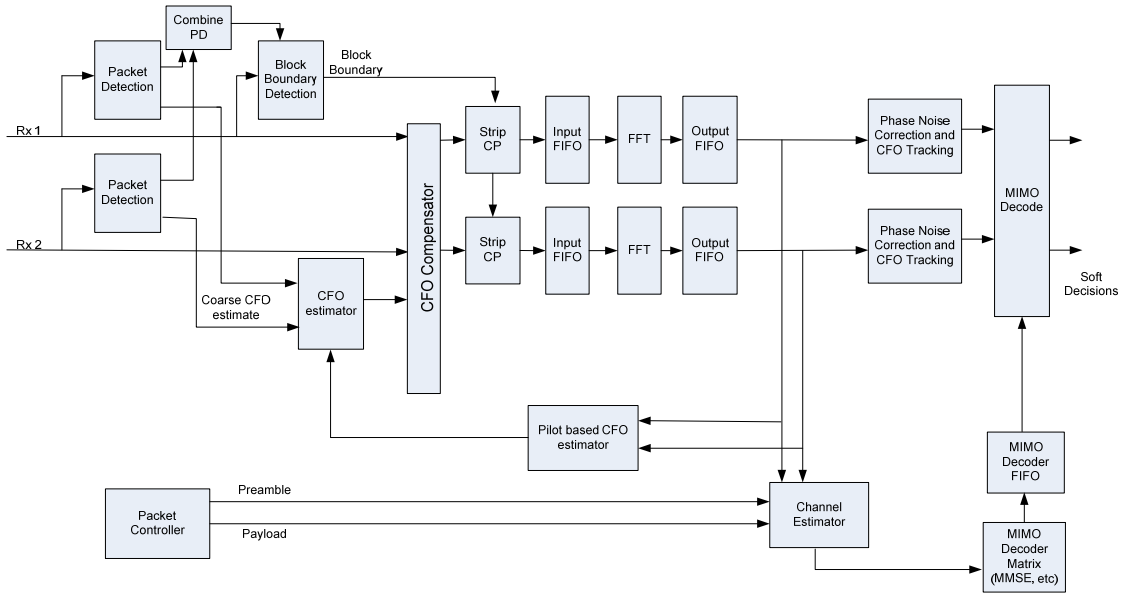
FlexOFDM receiver

The FlexOFDM receiver block diagram and System Generator for DSP implementation appear below. All the blocks up to the FFT process data at the baud rate or symbol rate. The FFT and all the blocks after it clock at the system’s clock rate. This allows resource sharing (folding) in the MIMO decoder and channel estimation blocks. The role of MIMO decoding is separated into two blocks:

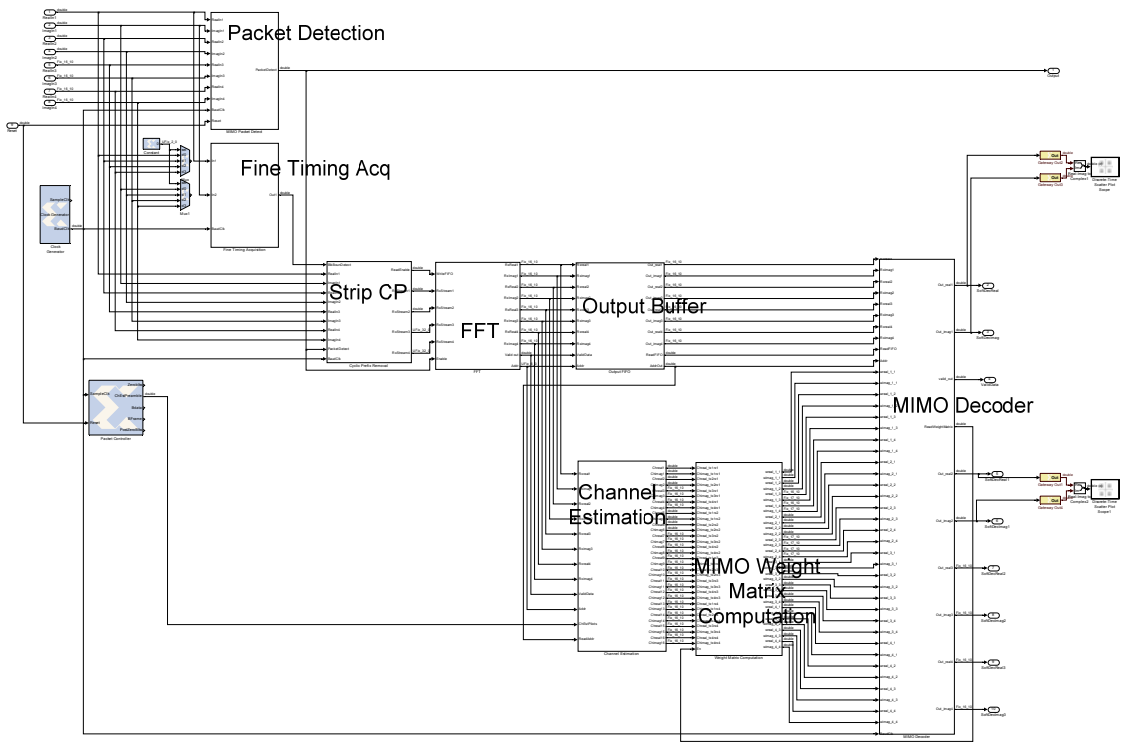
* MIMO weight matrix computation

* MIMO decoder

The MIMO weight matrix computation module operates on the channel matrix estimated by the channel estimation module and computes the weight matrix $W=H^H$, in the case of OSTBC. The MIMO decoder uses the weight matrix W and computes the soft estimate of the transmission symbols by performing $\hat{c}=Wr$, where \hat{c} is the estimate of the transmission symbols. The MIMO weight matrix computation module and MIMO decoder operate on one subcarrier at a time output from the output buffer. The figure below illustrates the System Generator for DSP implementation of the channel estimation block in a 2x2 MIMO FlexOFDM system.



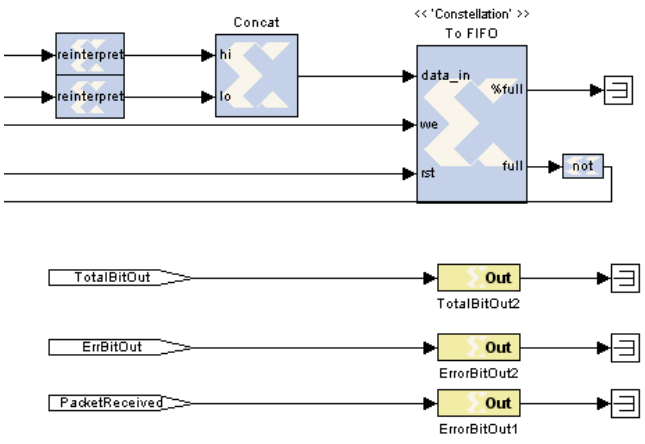
FlexOFDM receiver block diagram



MIMO OFDM decoder

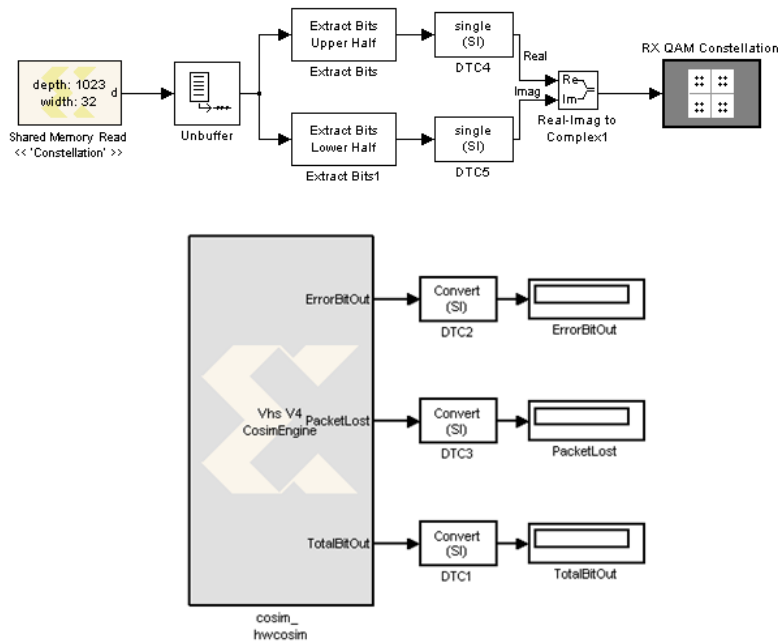
Hardware-in-the-loop co-simulation

It is now possible to visualize the received constellation in real-time using the co-simulation model in MATLAB. In the co-simulation subsystem, a shared FIFO and three gateway blocks in the FPGA model allows data exchange between the Simulink model and the VHS-ADC FPGA.



Shared memory and gateway blocks

Once the co-simulation model is started, it is possible to view in real-time the results of data being processed by the FPGA, for example the received constellation and the BER, i.e. the ratio of the number of bits incorrectly received to the total number of bits, sent in each received packet.



Shared memory and co-simulation blocks

Note

There is also an automatic gain control (AGC) section in the RX section model, but this section is not fully functional as of this release; thus, the AGC is not enabled when you run the demo. The gain is instead controlled by the host device (in the **OFDM_Demo_Config_Panel** dialog box, the **GAIN_CTRL** parameter is set to **GAIN_HOST_CTRL**). You can, however, modify and use the AGC section, but Lyrtech does not support problems caused by this section.

Receiver resource usage

FPGA	Input precision	FFT precision	FFT size	TX number	DSP48/ 18x18 mults	Slices	BRAMs
Virtex-4	<16,10>	12	64	2	188	20365	175

Requirements

To perform this example, you must meet the following requirements:

Hardware requirements

- * VHS-ADC (equipped with an SX55 FPGA and DC coupled) (×1)
- * VHS-DAC (equipped with an SX55 FPGA and DC coupled) (×1)

- * Revision C Quad Dual Band RF Transceiver (×2) that have undergone the proper hardware modifications (see note)
- * 2.4 GHz antenna (×4)
- * SMA–MMCX cable (×14)
- * SMA–SMA cable (×4)
- * GPIO cable (×2)
- * ADACSync (×2)
- * JTAG probe from Xilinx (×1) (Only if you want to use the ChipScope Pro analyzer)

Note

Some hardware modifications are necessary on revision C of the Quad Dual Band RF Transceiver, which involve modifying the RSSI dynamic range, supporting an external clock, and removing carrier leakage. This is handled through an RMA.

Software requirements

- * MATLAB r2008a+
- * ISE Foundation 10.1.03 IP update 3
- * ChipScope Pro 10.1
- * System Generator for DSP 10.1.3.1386
- * OFDM_Demo_Config_Panel
- * VHS control utility

Setup

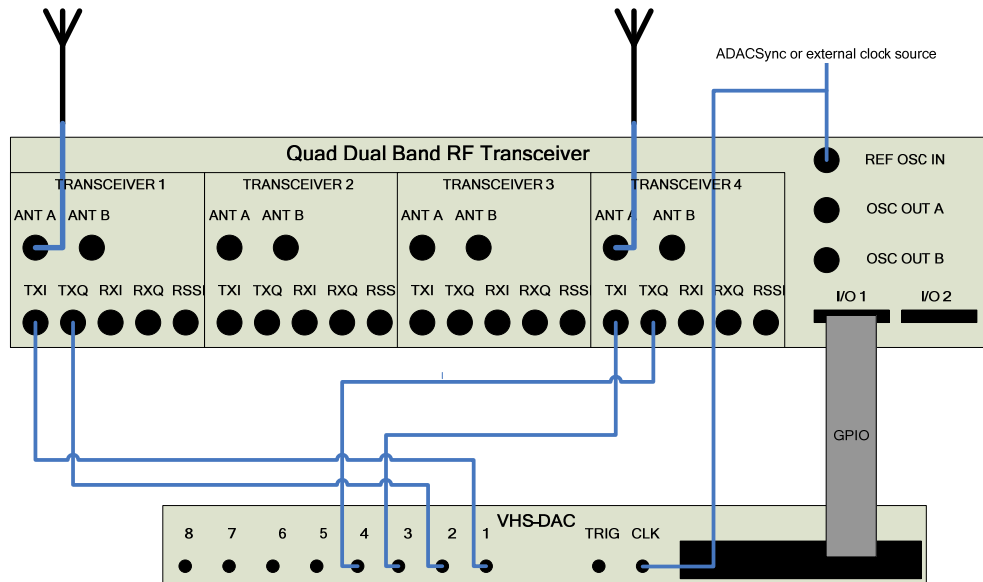
TX section

1. Connect an antenna to the **TRANSCEIVER 1 ANT A** connector of your Quad Dual Band RF Transceiver.
2. Connect an antenna to the **TRANSCEIVER 4 ANT A** connector of your Quad Dual Band RF Transceiver.
3. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 1 TXI** connector of your Quad Dual Band RF Transceiver.
4. Connect the opposite end of the SMA–MMCX cable to channel 1 of your VHS-DAC.
5. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 1 TXQ** connector of your Quad Dual Band RF Transceiver.
6. Connect the opposite end of the SMA–MMCX cable to channel 2 of your VHS-DAC.
7. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 4 TXI** connector of your Quad Dual Band RF Transceiver.
8. Connect the opposite end of the SMA–MMCX cable to channel 3 of your VHS-DAC.
9. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 4 TXQ** connector of your Quad Dual Band RF Transceiver.
10. Connect the opposite end of the SMA–MMCX cable to channel 4 of your VHS-DAC.
11. Connect a flat GPIO cable from the VHS-DAC to the **I/O 1** connector of your Quad Dual Band RF Transceiver.
12. Connect a cable between the **REF OSC IN** connector of your Quad Dual Band RF Transceiver and your clock source.

Note

You can use two ADACSyncs in your design (one RX ADACSync, one TX ADACSync) or you can use a single external clock at the RX and TX. Make sure that the RX and TX clocks are synchronized for consistent results. The VHS-ADC and VHS-DAC's operating frequencies are 40 MHz. To correctly configure the ADACSync, refer to its [user's guide](#).

13. Connect an SMA–MMCX cable to the clock source.
14. Connect the other end of the SMA–MMCX cable to the **CLK** connector of your VHS-DAC.
15. Insert the VHS-DAC into your cPCI chassis.



TX section connection diagram

RX section

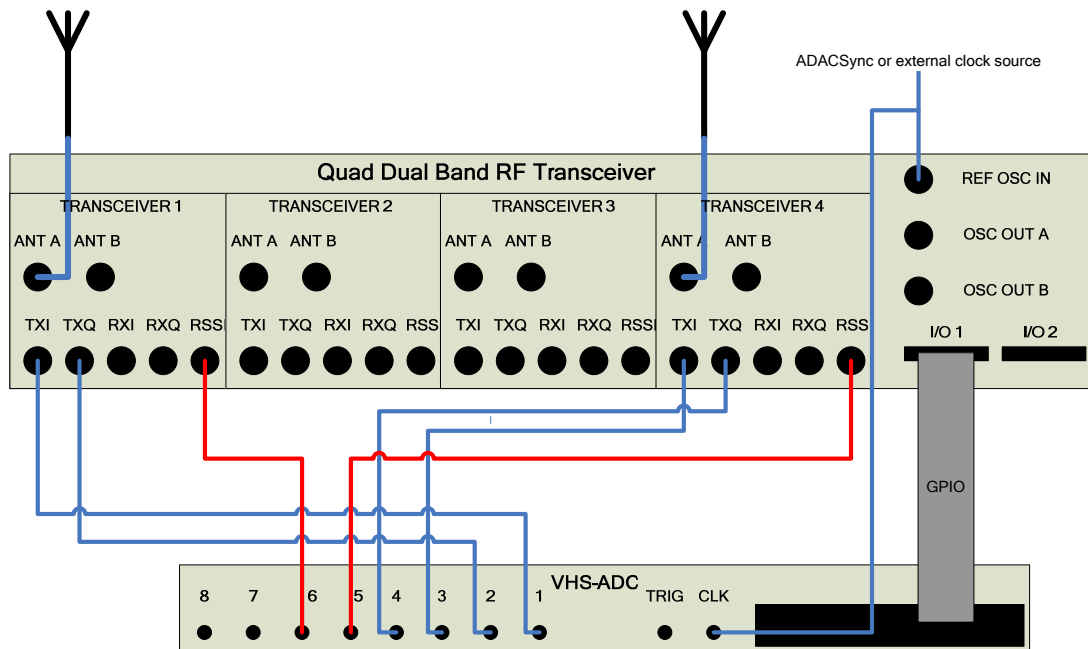
1. Connect an antenna to the **TRANSCEIVER 1 ANT A** connector of your second Quad Dual Band RF Transceiver.
2. Connect an antenna to the **TRANSCEIVER 4 ANT A** connector of your second Quad Dual Band RF Transceiver.
3. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 1 RXI** connector of your second Quad Dual Band RF Transceiver.
4. Connect the opposite end of the SMA–MMCX cable to channel 1 of your VHS-ADC.
5. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 1 RXQ** connector of your second Quad Dual Band RF Transceiver.
6. Connect the opposite end of the SMA–MMCX cable to channel 2 of your VHS-ADC.
7. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 4 RXI** connector of your second Quad Dual Band RF Transceiver.
8. Connect the opposite end of the SMA–MMCX cable to channel 3 of your VHS-ADC.
9. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 4 RXQ** connector of your second Quad Dual Band RF Transceiver.
10. Connect the opposite end of the SMA–MMCX cable to channel 4 of your VHS-ADC.
11. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 1 RSSI** connector of your second Quad Dual Band RF Transceiver.
12. Connect the opposite end of the SMA–MMCX cable to channel 5 of your VHS-ADC.

13. Connect one end of an SMA–MMCX cable to the **TRANSCEIVER 4 RSSI** connector of your second Quad Dual Band RF Transceiver.
14. Connect the opposite end of the SMA–MMCX cable to channel 6 of your VHS-ADC.
15. Connect a flat GPIO cable from the VHS-ADC to the **I/O 1** connector of your second Quad Dual Band RF Transceiver.
16. Connect a cable between the **REF OSC IN** connector of your Quad Dual Band RF Transceiver and your clock source.

Note

You can use two ADACSyncs in your design (one RX ADACSync, one TX ADACSync) or you can use a single external clock at the RX and TX. Make sure that the RX and TX clocks are synchronized for consistent results. The VHS-ADC and VHS-DAC's operating frequencies are 40 MHz. To correctly configure the ADACSync, refer to its [user's guide](#).

17. Connect an SMA–MMCX cable to the clock source.
18. Connect the other end of the SMA–MMCX cable to the **CLK** connector of your VHS-ADC.
19. If you want to use the ChipScope Pro analyzer, connect the JTAG probe to the VHS-ADC's JTAG header.
20. Insert the VHS-ADC into your cPCI chassis.



RX section connection diagram

Note

The TX and RX sections must be 42 inches from each other. In each section, the antennas must be 12 inches from each other.

Testing the example

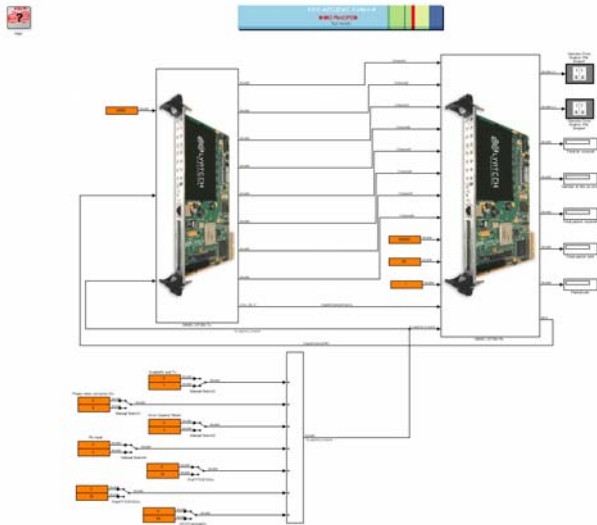
Real-time hardware implementation

Creating a TX bitstream

Note

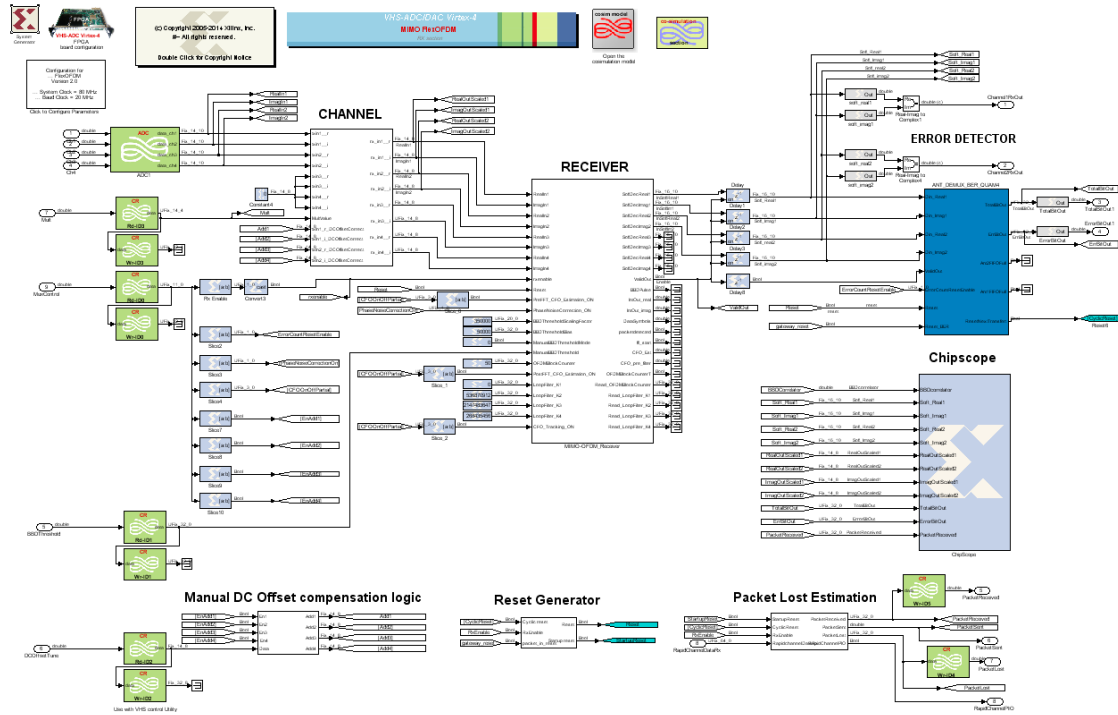
You can save time by using the supplied bitstream (*vhs_adac_virtex4_ff1148_ioring_TX.bit*) located in *ADPLOC\matlab\r2008a\VHS-V4\demos\vhsadac_v4_ofdm_files*, instead of creating your own.

1. Click the banner at the top of this Help file to open the example model.



MIMO FlexOFDM top model

2. Double-click **TX section**.



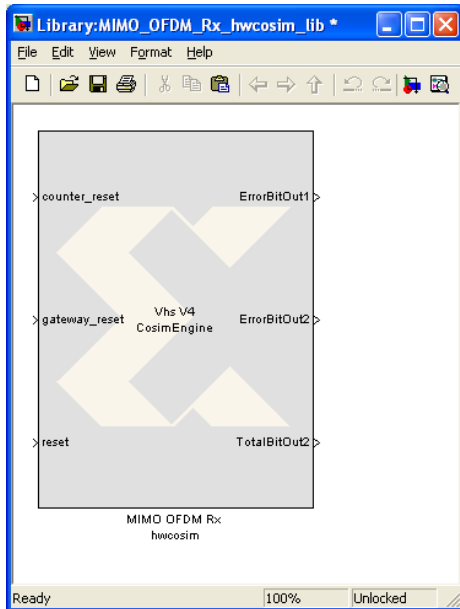
RX section

3. Double-click the **System Generator** block.

Two dialog boxes open—the **System Generator** and the **FPGA configuration** dialog boxes.

4. In the **System Generator** dialog box, click **Generate**.

The bitstream configuration file is generated in the folder specified in the **Target directory** text box. A copy of the bitstream is also saved in the current MATLAB folder. A HIL co-simulation model also opens at the end of the generation containing the MIMO OFDM RX block. This block contains all the information about the shared FIFOs in the RX FPGA model.



HIL co-simulation block created after bitstream compilation

Configuring the VHS-DAC (TX section)

Note

You must configure the VHS-DAC from the TX section chassis containing the development platform.

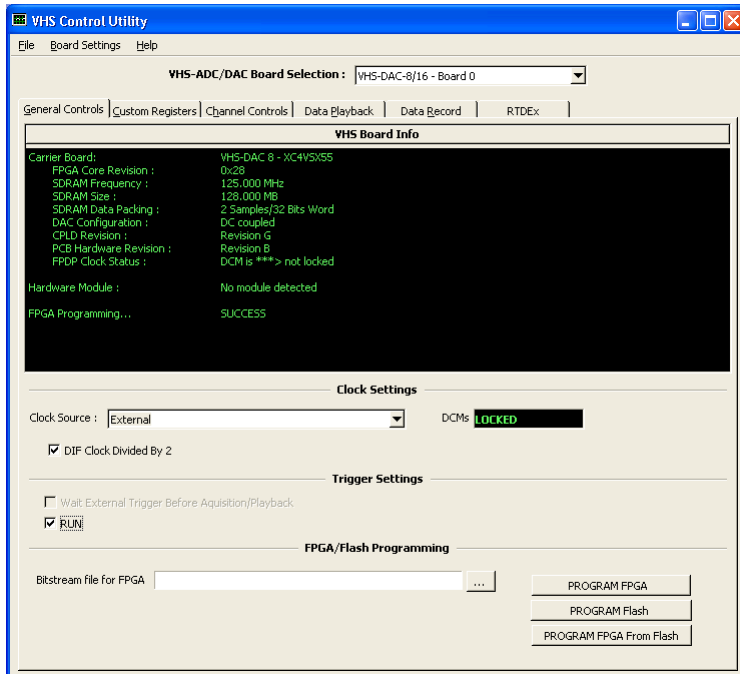
1. Copy the previously generated TX bitstream to the chassis containing your VHS-DAC.
2. Make sure that your clock source is correctly connected to the Quad Dual Band RF Transceiver's **REF OSC IN** connector and that it is functioning.

Note

If you are using one or several ADACSyncs, make sure that you program them to generate the clock. Refer to the ADACSync's [user's guide](#) for details.

3. On the Windows **Start** menu, point to **All Programs, Lyrtech, Host SDK, cPCI Platforms**, and then click **VHS Control Utility**.

The VHS control utility starts.



VHS control utility (TX section)

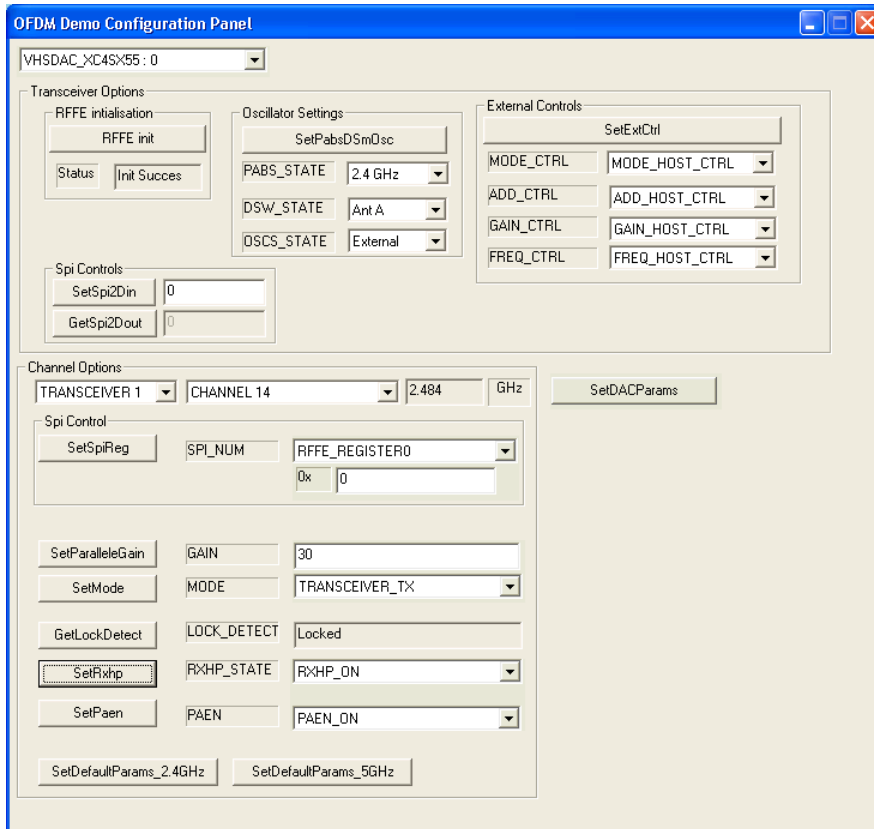
4. On the **VHS-ADC/DAC Board Selection** list of the **General Controls** tab, select **VHS-DAC-8/16—Board 0**.
5. In the **Bitstream file for FPGA** text box, specify the previously generated TX bitstream that you created in [Creating a TX bitstream](#).
6. Click **Program FPGA**.
7. On the **Clock Source** list, select **External Front Panel**.
8. Select the **RUN** check box.

Configuring the TX Quad Dual Band RF Transceiver

Note

Perform the following procedure on the chassis containing the VHS-DAC of the TX section.

1. Run *OFDM_Demo_ConfigPanel.exe* located in *ADPLOC\matlab\2008a\VHS-V4\demos\vhsadac_v4_ofdm_files*.



OFDM Demo Configuration Panel dialog box (TX section)

2. In the above dialog box, click **SetDacParams**. In the **Transceiver Options** group, click **RFFE init**.

Make sure that you receive an *Init success* message before continuing.

3. On the **PABS_STATE** list, select **2.4 GHz**.
4. On the **DSW_STATE** list, select **Ant A**.
5. On the **OSCS_STATE** list, select **External**.

Note

These values are already hardcoded in the FPGA custom logic.

6. Click **SetPabsDSmOsc**.
7. In the **Channel Options** group, select **TRANSCIEVER 1**, and then **CHANNEL 14**.
8. Click **SetDefaultParams_2.4GHz**.
9. On the **MODE** list, select **TRANSCEIVER_SPI_RESET**.
10. Click **SetMode**.
11. On the **MODE** list, select **TRANSCEIVER_TX**.
12. Click **SetMode**.
13. Click **GetLockDetect**.

Make sure that the state is *Locked* before proceeding.

14. In the **GAIN** text box, specify **30**, and then click **SetParalleleGain**.
15. On the **PAEN** list, select **PAEN_ON**.
16. Click **SetPaen**.

17. Select **TRANSCEIVER 4**, and then select **CHANNEL 14**.
18. Click **SetDefaultParams_2.4GHz**.
19. On the **MODE** list, select **TRANCEIVER_TX**.
20. Click **SetMode**.
21. Click **GetLockDetect**.
Make sure that the state is *Locked* before proceeding.
22. In the **GAIN** text box, type *30*, and then click **SetParalleleGain**.
23. On the **PAEN** list, select **PAEN_ON**.
24. Click **SetPaen**.
25. In the VHS control utility, on the **Custom Registers** tab, in the **Write Field** text box of the **Custom Register 1** group, specify *20000*, and then click **Write**.
26. In the **Write Field** text box of the **Custom Register 0** group, specify *1*, and then click **Write**.
The transmitter starts.

Configuring the VHS-ADC (RX section)

Note

You must configure the VHS-ADC from the RX section chassis containing the development platform.

1. Copy the previously generated RX bitstream to the RX section chassis.

Note

If you want to use the ChipScope Pro analyzer, also copy the previously generated *.cdc* file to the RX section chassis.

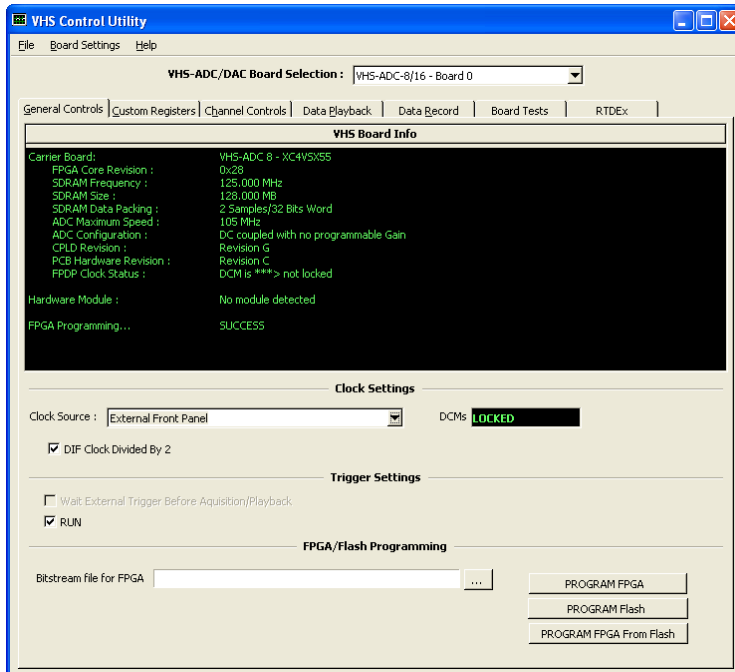
2. Make sure that your clock source is properly connected to the Quad Dual Band RF Transceiver's **REF OSC IN** connector and functioning.

Note

If you are using one or several ADACSyncs, make sure that you program them to generate the clock. Refer to the ADACSync's [user's guide](#) for details.

3. If you are using the ChipScope Pro analyzer, make sure that your JTAG probe is properly connected to the VHS-ADC.
4. On the Windows **Start** menu, point to **All Programs, Lyrtech, Host SDK, cPCI Platforms**, and then click **VHS Control Utility**.

The VHS control utility starts.



VHS control utility (RX section)

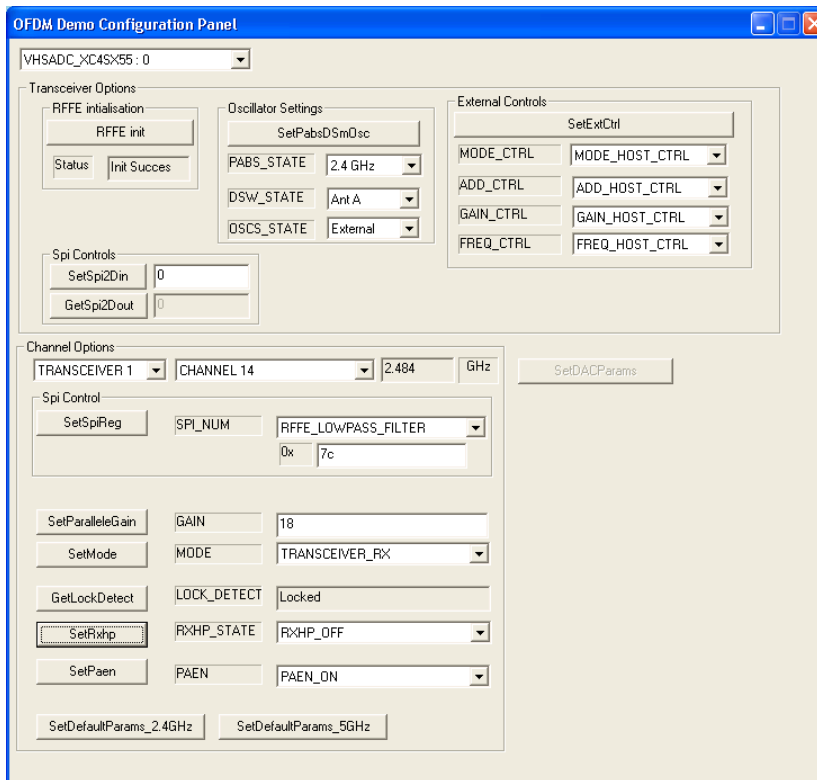
5. On the **VHS-ADC/DAC Board Selection** list of the **General Controls** tab, select **VHS-ADC-8/16—Board 0**.
6. In the **Bitstream file for FPGA** text box, specify the RX bitstream that you generated in [Creating an RX bitstream](#).
7. Click **Program FPGA**.
8. On the **Clock Source** list, select **External Front Panel**.
9. Select the **RUN** check box.

Configuring the RX Quad Dual Band RF Transceiver

Note

Perform the following procedure on the chassis containing the VHS-ADC of the RX section.

1. Run *OFDM_Demo_ConfigPanel.exe* located in *ADPLOC\matlab\r2008a\VHS-V4\demos\vhسادc_v4_ofdm_files*.
2. In the **Transceiver Options** group, click **RFFE init**.
Make sure that you receive an *Init success* message before proceeding.



OFDM Demo Configuration Panel dialog box (RX section)

3. On the **PABS_STAT** list, select **2.4 GHz**.
4. On the **DSW_STATE** list, select **Ant A**.
5. On the **OSCS_STATE** list, select **External**.

Note

These values are hardcoded in the FPGA's custom logic.

6. Click **SetPabsDSmOsc**.
7. In the **Channel Options** group, select **TRANSCIEVER 1**, and then **CHANNEL 14**.
8. Click **SetDefaultParams_2.4GHz**.
9. On the **MODE** list, select **TRANSCIEVER_SPI_RESET**.
10. Click **SetMode**.
11. On the **MODE** list, select **TRANSCIEVER_RX**.
12. Click **SetMode**.
13. On the **RXHP_STATE** list, select **RXHP_OFF**.
14. Click **SetRxhp**.
15. Click **GetLockDetect**.
Make sure that the state is *Locked* before proceeding.
16. On the **SetSpiReg** list, select **RFFE_LOWPASS_FILTER**, and then specify **7C** in the corresponding text box.
17. Click **SetSpiReg**.
18. In the **GAIN** text box, specify the maximum value before the OVR LED lights, and then click **SetParalleleGain**.
19. In the **Channel Options** group, select **TRANSCIEVER 4**.

20. Select **CHANNEL 14**.
21. Click **SetDefaultParams_2.4GHz**.
22. On the **MODE** list, select **TRANSCEIVER_RX**.
23. Click **SetMode**.
24. On the **RXHP_STATE** list, select **RXHP_OFF**.
25. Click **SetRxhp**.
26. Click **GetLockDetect**.
27. On the **SetSpiReg** list, select **RFFE_LOWPASS_FILTER**, and then specify **7C** in the corresponding text box.
28. Click **SetSpiReg**.
29. In the **GAIN** text box, specify the maximum value before the OVR LED lights, and then click **SetParalleleGain**.
30. In the VHS control utility, on the **Custom Registers** tab, in the **Write Field** text box of the **Custom Register 1** group, specify **500000**, and then click **Write**.

Note

Custom Register 1 determines the value of the BBD threshold. The BBD threshold must be between **500000** and **1000000**. A value under 500,000 may create an invalid BER and a value over 1,000,000 may stop the HIL co-simulation because there is not enough valid data to feed the shared FIFO, making it time out.

31. In the VHS control utility, on the **Custom Registers** tab, in the **Write Field** text box of the **Custom Register 2** group, specify **1**, and then click **Write**.
32. In the VHS control utility, on the **Custom Registers** tab, in the **Write Field** text box of the **Custom Register 3** group, specify **16**, and then click **Write**.
33. In the VHS control utility, on the **Custom Registers** tab, in the **Write Field** text box of the **Custom Register 0** group, specify **9**, and then click **Write**.

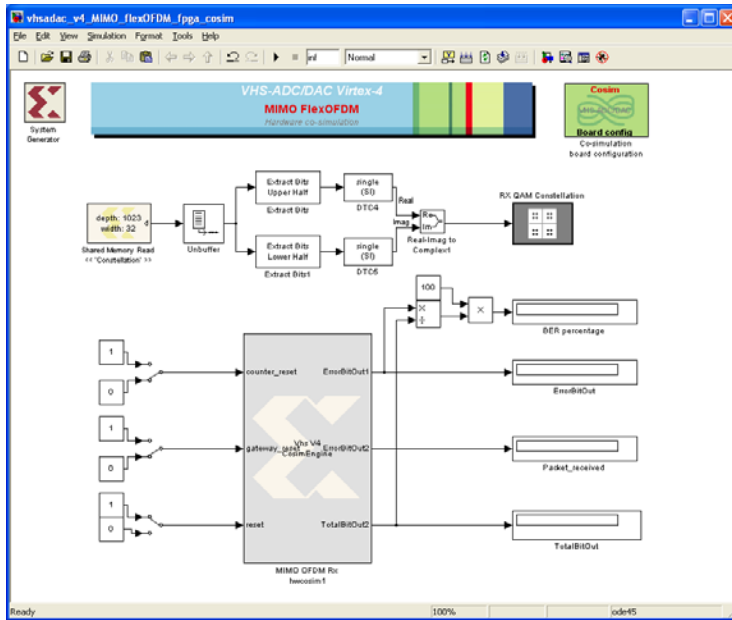
Running the HIL co-simulation model

Note

Perform the following procedure on the chassis containing the VHS-ADC of the RX section.

34. In MATLAB, open the MIMO FlexOFDM top model, and then click **Open co-simulation model** block.

The HIL co-simulation model opens.



HIL co-simulation model

35. Replace the **MIMO OFDM Rx hwcosim** block by the one that you generated previously.

See [Creating an RX bitstream](#).

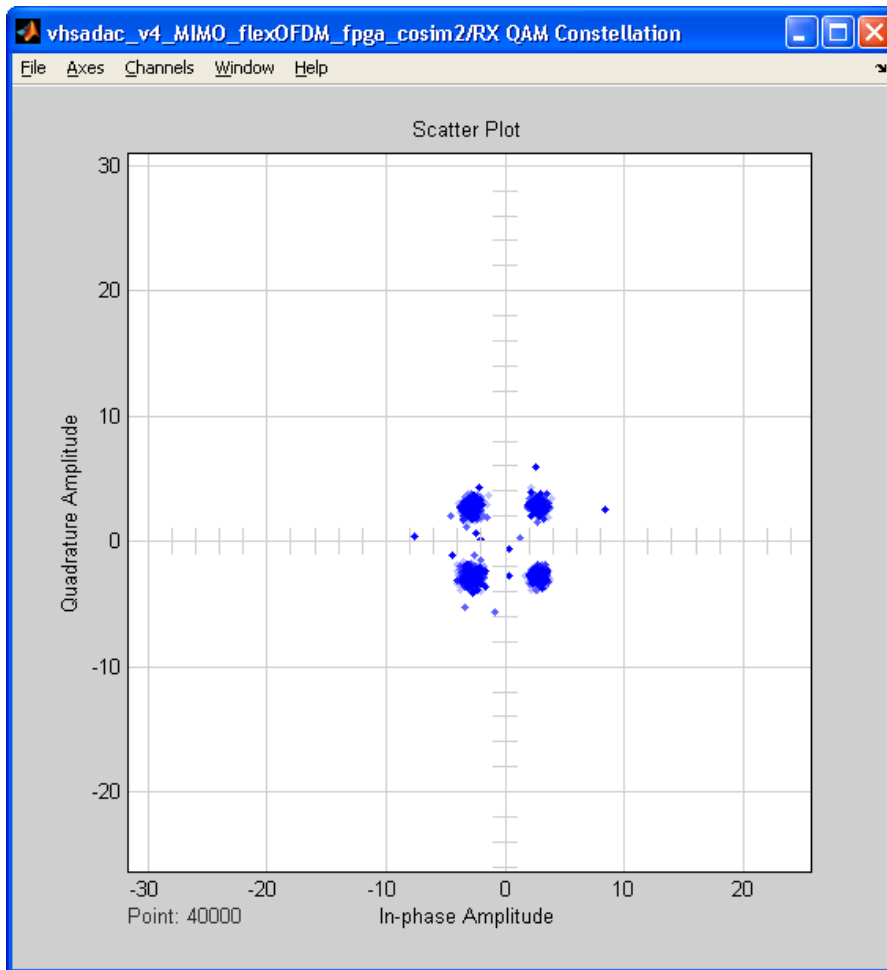
36. On the model's **Simulation** menu, click **Start**.

Note

You **must** always run the HIL co-simulation model on the CPU of the cPCI chassis. Running it on a remote computer generates an error.

Expected results

The following are the expected results of the example. You can observe the received constellation on the **RX QAM constellation** scope.



RX QAM constellation

You can also visualize the BER percentage, the total bits sent, the number of error bits, and the number of packets received on the various displays. The expected BER must be between 0.5 % and 4.0 %. See [Fine adjustment using ChipScope Pro](#) to help you reach the lowest possible BER.

If you use ChipScope Pro to acquire data from a valid transmission

1. Start ChipScope Pro 10.1.
2. Make sure that your JTAG probe is connected to the RX section's VHS-ADC.
3. Double-click **Open Cable/Search JTAG chain**.
There should be two XC4VSX55 devices and one XC2C256 device.
4. Click **OK**.
5. On the **File** menu, click **Import**.
6. Select the previously generated *flexofdmadctorx_v3_ChipScope Pro1.cdc* file.
7. Click **OK**.
8. In the **New Project** window, open the **Trigger Setup** window.
9. In **Trigger Condition Equation**, select **M0** for the BBDcorrelator signal.

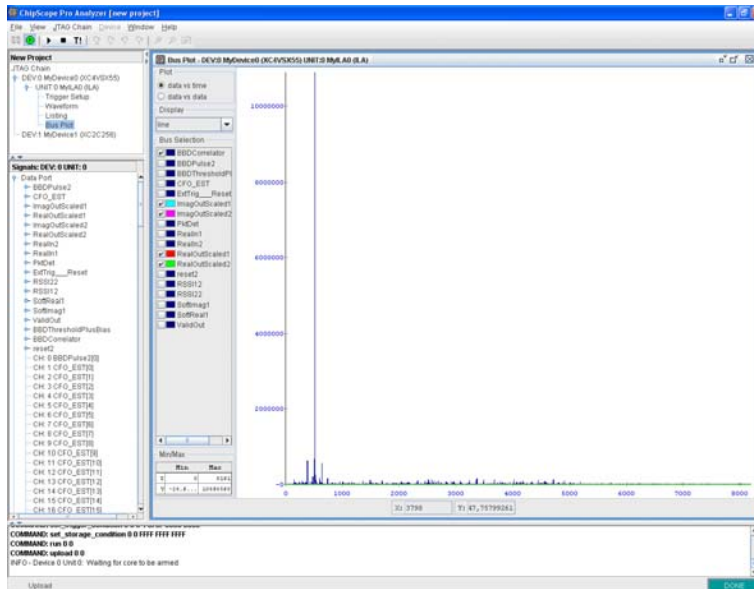
10. In the **Match** group, select the column **M0:BBDCorrelator XXXX XXXX XXXX 1XXX XXXX XXXX XXXX XXXX** to configure a trigger threshold so that it is equal to or greater than **524288**.

11. In ChipScope Pro, click the play button.

12. In the **New Project** window, open the **Bus Plot** window.

13. In the **Bus Plot** window, select the **BBDCorrelator** signal.

You should see a signal similar to the following peaking around 512. The peak is the correlation product between the signals received at each antenna and the preamble sequence of the RX transceiver. The peak value can vary from one transmission to the other



Correlation

14. In the **Bus Plot** window, release the **BBDCorrelator** signal.

15. In the **Plot** group of the **Bus Plot** window, select the **data vs. data** option.

16. In the **Bus Selection** group, select the **SoftReal1** check box for x values and the **SoftImag1** check box for y values.

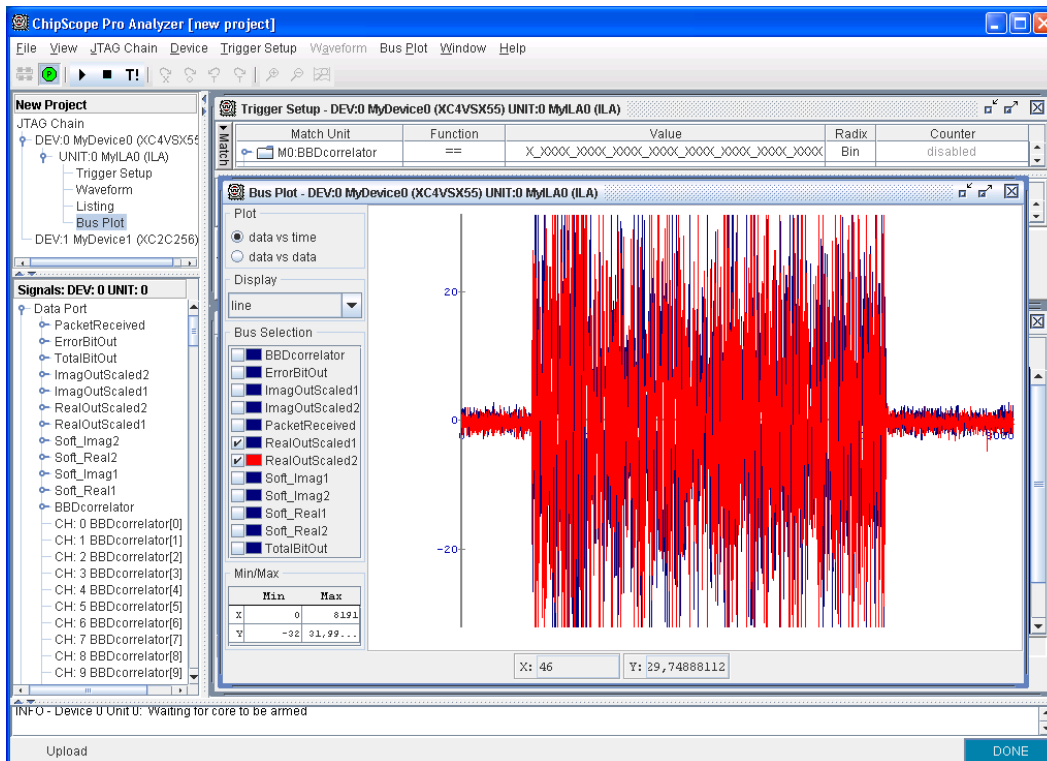
You can select lines or points to visualize the constellation on the **Display** list.

Note

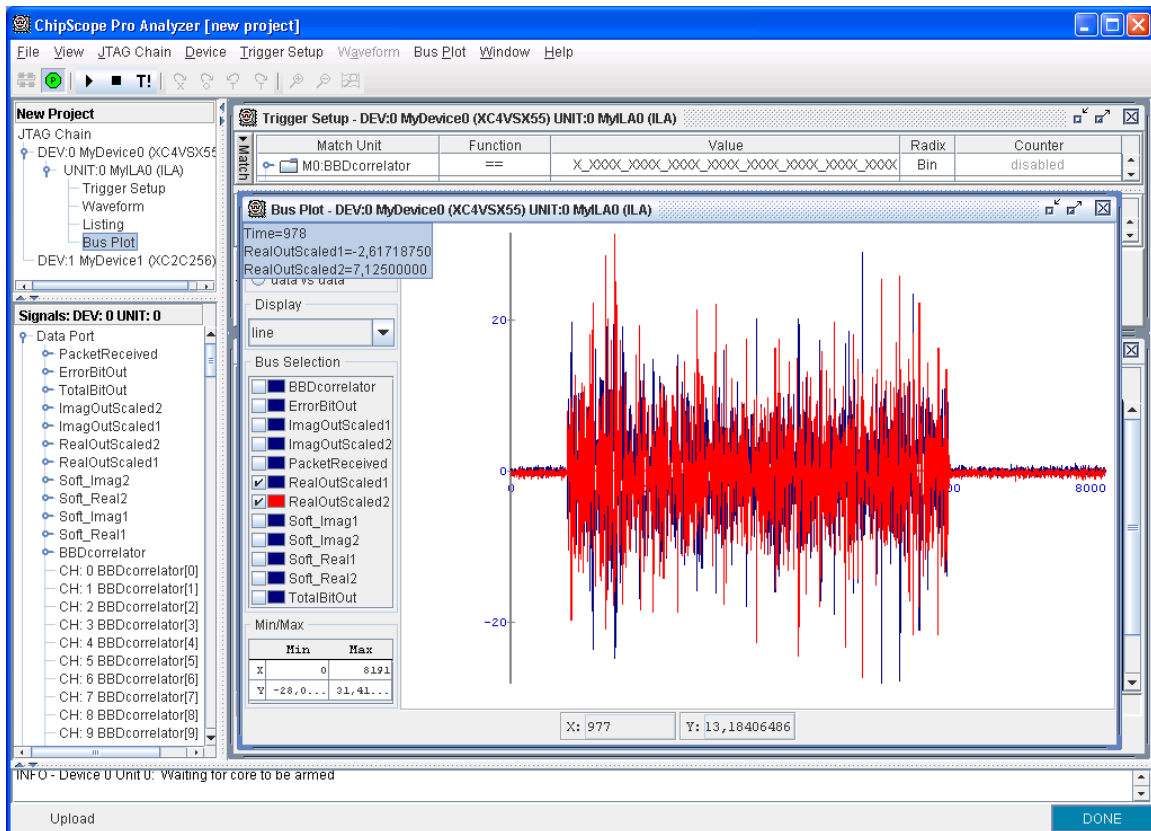
Zeros in the center of the constellation are intentionally there to show the amount of noise.

3. Click the play button.

You should see the two signals similar to the following with a burst of data in each of them. Make sure that the maximum values of the bursts are close to 32 but are not saturated. If the signals are too low or too high, adjust the corresponding transceiver gain value to increase or decrease the amplitude of the signals until you reach the appropriate gain values.



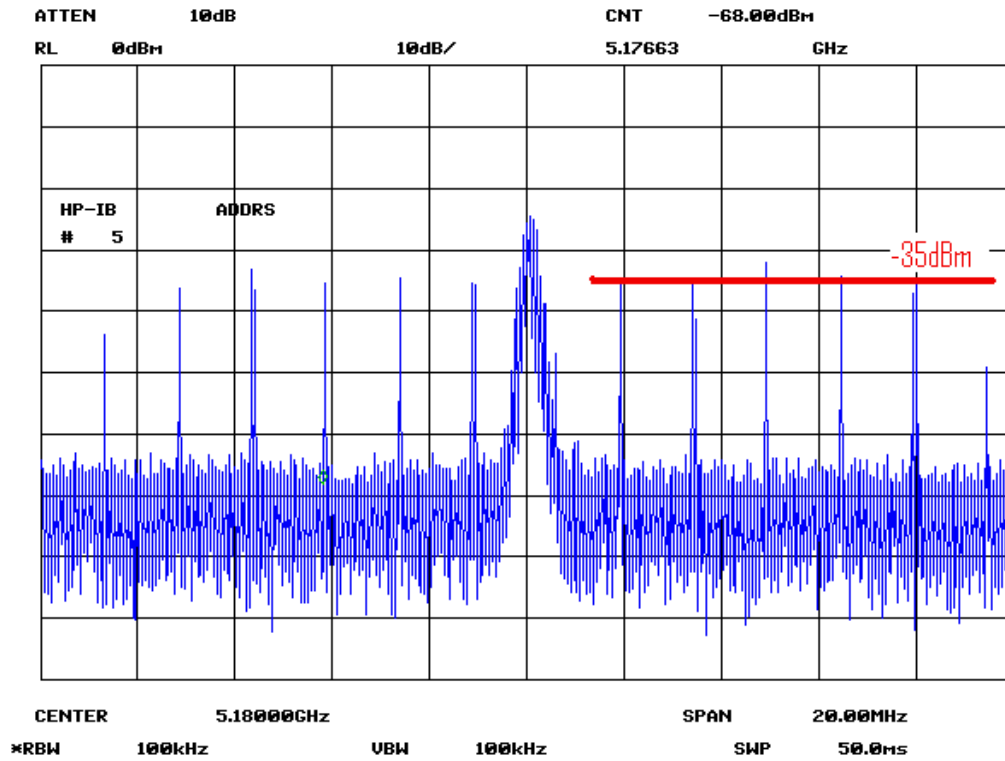
RealOutAcaled1 and RealOutScaled2 signals saturated; the gain is too high



RealOutScaled1 and RealOutScaled2 signals correctly configured

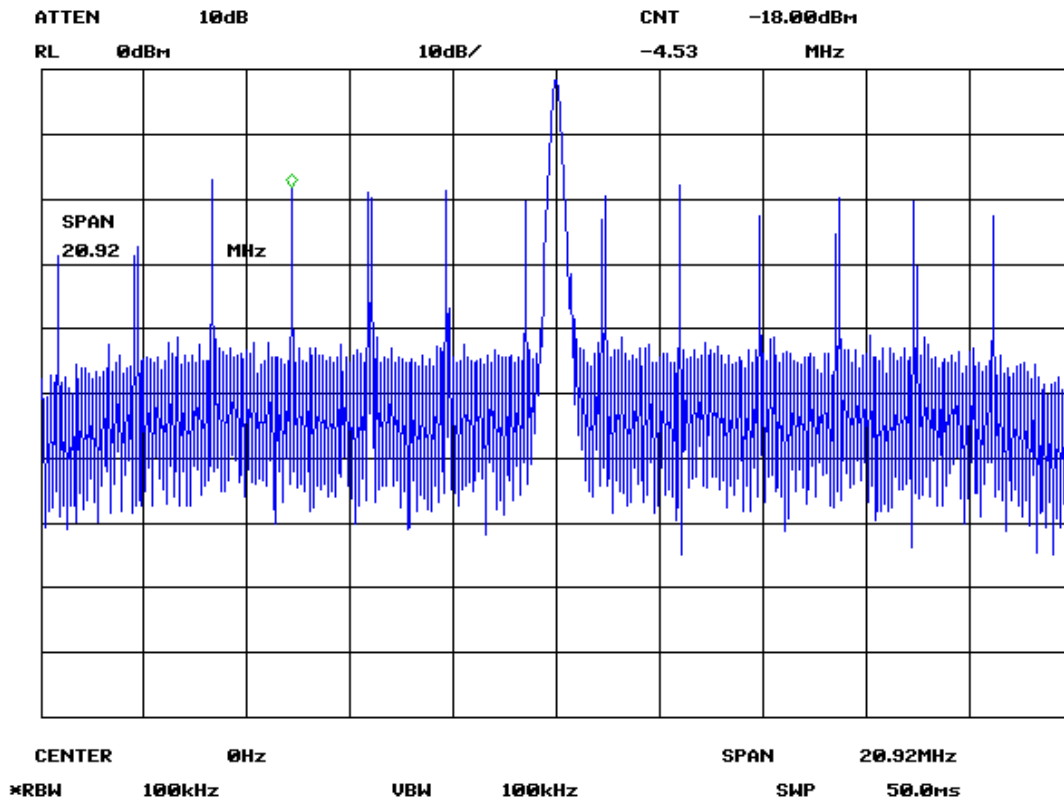
Potential problems affecting transmissions

- * To obtain a clean constellation, make sure that the frequency band is not polluted by other Wi-Fi signals. To do so, stop the TX section and visualize the RXI or RXQ signals of the RX section on a spectrum analyzer, over a range of 50 MHz. Verify that there are no bursts in the baseband signal. Disable devices functioning in this range of frequencies—for example laptop computers—or change Wi-Fi channels. We recommend that you use CHANNEL 14, which should be less noisy, but you can use any channel (both TX and RX transceivers must use the same channel, however).
- * Make sure that both TX transceivers are transmitting at the same level (approximately – 35 dBm). To modify the level adjust the **GAIN** parameter in the TX section's **OFDM Demo Configuration Panel** dialog box. Visualize the TX section's TXI or TXQ signals with a spectrum analyzer. Take care not to leave ANT A and ANT B without load when the Quad Dual Band RF Transceiver is on. The spectrum analyzer should display a signal resembling the following around 2.4 GHz.



OFDM signal (TX ANT A)

- * Make sure that both RX transceivers are receiving the signal by visualizing the RXI and RXQ signals of the transceivers with a spectrum analyzer (see the following figure).



OFDM signal (RX RXI or RXQ)

- * Make sure that connections are exactly as illustrated in the above connection diagrams. If the I and Q signals are inverted, results are affected.
- * If the correlation threshold is too high or too low, you are unable to visualize the constellation. The threshold must be configured at a low enough level to be triggered by the peak at 512, but still be high enough not to be triggered by lower peaks. You can miss the demodulation if the threshold is not properly configured.