

Aurora link with Typhoon via SFP

PN112 | Posted on April 6, 2026 | Updated on April 7, 2026



François LEDENT

Development Engineer

imperix • in

Table of Contents

- [Case study](#)
- [Required software](#)
- [Downloads](#)
- [Typhoon HIL application](#)
- [Communication chain](#)
 - [Overview](#)
 - [Vivado project](#)
 - [Modules/IPs description](#)
 - [Aurora parameters](#)
- [Experimental validation](#)
 - [Physical setup](#)
 - [Software-side setup](#)
 - [Real-time monitoring](#)

This page presents a practical example of Aurora communication with Typhoon HIL simulators, specifically the HIL101, HIL404, HIL506, and HIL606. It provides a ready-to-use user application, along with an FPGA design specifically configured for interfacing Typhoon HIL simulators.

To demonstrate the setup, this page also includes pre-configured Typhoon Schematic Editor and HIL SCADA files implementing a simple loopback example. In this example, a three-phase sine wave is transmitted from the imperix controller, multiplied by two in the Typhoon HIL, and sent back to the controller.

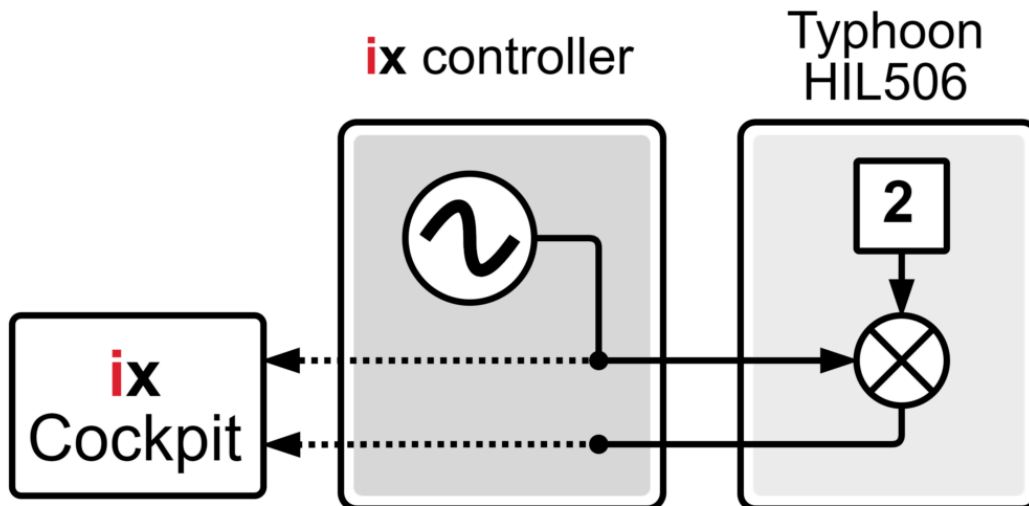
For broader technical background, a general introduction to [SFP communication with third-party devices](#) is available. That page covers fundamental SFP considerations, describes the full communication chain, and provides guidance on implementing the imperix-side drivers.

Similar pages are available for OPAL-RT and Plexim devices in [Aurora link with OPAL-RT via SFP](#) and [Aurora link with Plexim via SFP](#).

Case study

This case study demonstrates a straightforward signal processing loop:

- The controller generates a three-phase sine wave and transmits it to the HIL506 via SFP.
- Upon receipt, the HIL506 applies a gain of two to the three signals – doubling the amplitude of the sine wave – and returns them to the controller, also via SFP.
- Finally, the original transmitted values and the received return signals are compared in real-time in Cockpit, showing the proper operation of the system.



Required software

- **Vivado Design Suite** (version **2022.1** is recommended)
The [Xilinx installation page](#) details the installation procedure.
- **FPGA sandbox template 3.10** or later.
Available on the [FPGA download](#) page.
- **C++ or ACG SDK version 2024.3** or later.
Available on the [SDK download](#) page.

On the third-party side, this project has been tested with a **Typhoon HIL506**, **Typhoon HIL Control Center 2026.1 SP1**.

Downloads

As explained in the [setup overview](#), the SFP communication requires the three following software parts:

- The **user application**, running in the imperix controller's CPU and provided as a Simulink or PLECS script.
- The **FPGA bitstream**, loaded in the imperix controller's FPGA and provided as generation scripts.

- The **third-party application**, here running in the Typhoon simulator and provided as an archive containing the Typhoon Schematic Editor and Typhoon HIL SCADA files.

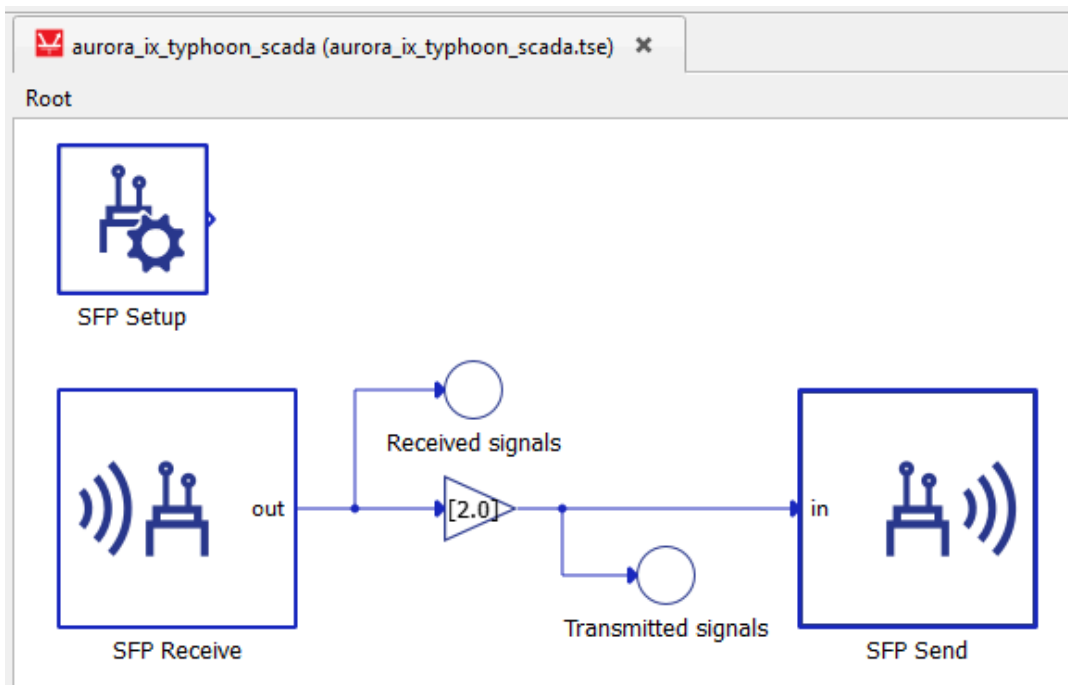
User application	FPGA bitstream	HIL506 application
aurora_ix_template.slx aurora_ix_template.plecs	aurora_ix_typhoon_gen_scripts.zip	aurora_ix_typhoon_files.zip

To enable communication with imperix controllers, the **SFP header must be deactivated** in the Typhoon simulator. This is done by selecting the **Disable header** checkbox within the **SSL** tab of the **SFP Setup** block.

Typhoon HIL application

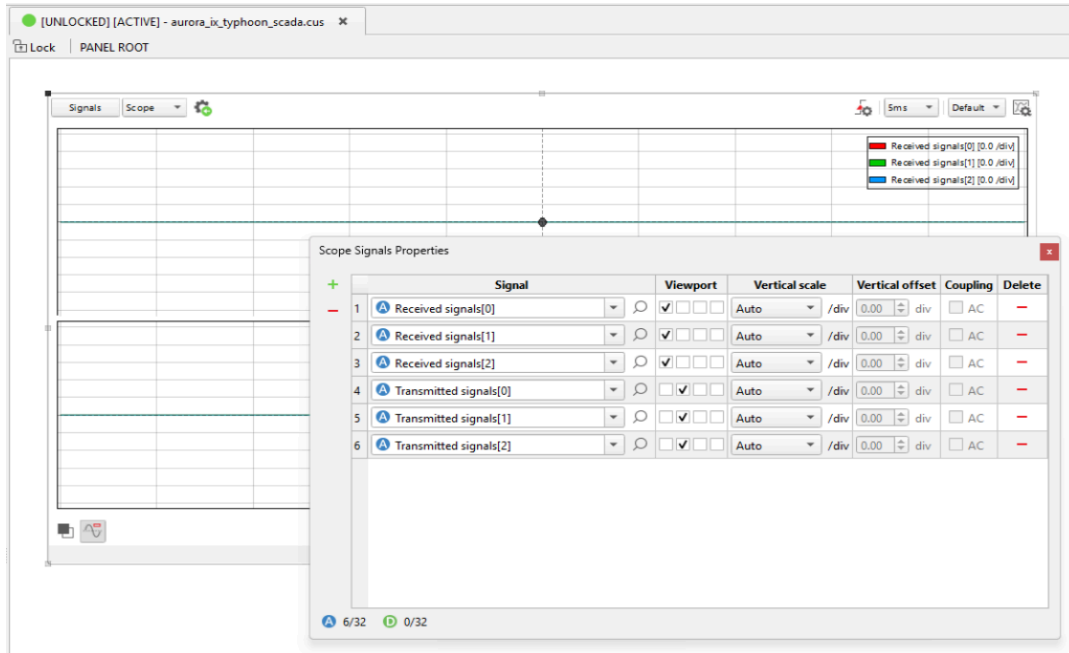
The application running in the HIL506 essentially contains an **SFP Receive** block to receive signals from the imperix controller via SFP, a **gain** of 2, and an **SFP Send** block to send the data back to the controller. It also contains the mandatory **SFP Setup** block to configure the SFP interface.

The **SFP 1** port is selected in the SFP Receive and SFP Send blocks, and the **header is disabled** from the SFP Setup block. The values received through SFP are outputted by the SFP Receive block as a vector of three floating-point elements. The execution rate of all blocks is set to 2 us.



To build and load the model on the HIL506, open the *aurora_ix_typhoon_sch_editor.tse* design in the [Schematic Editor](#) and click the **Compile and (re)load model in HIL SCADA** icon. Once the compilation is finished, [HIL SCADA](#) will launch automatically and the code will be loaded onto the simulator.

In HIL SCADA, open the provided *aurora_ix_typhoon_hil_scada.cus* panel and start the simulation. At this point, the code is running on the HIL506; data exchanged over SFP can then be monitored in the HIL SCADA scope (which will display only zeros if the imperix controller is not yet configured).

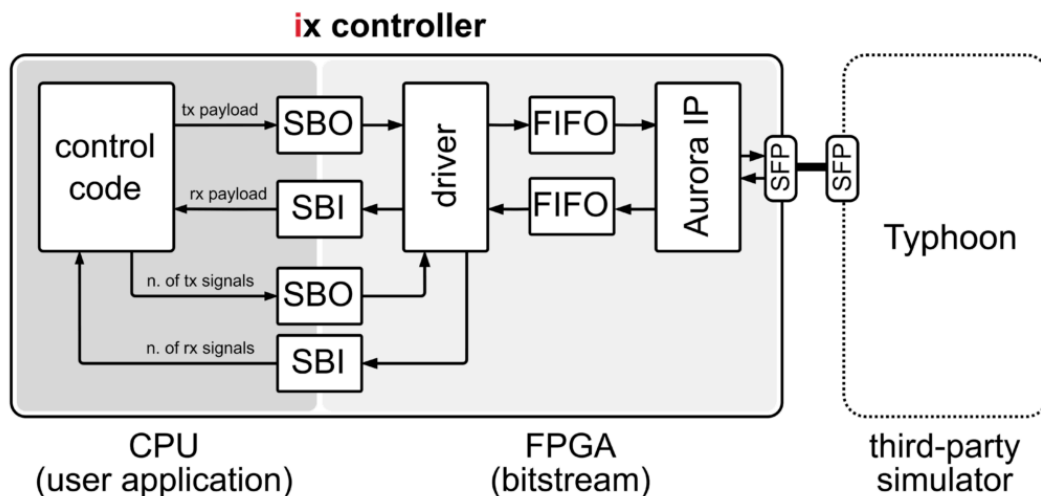


For further details on the Typhoon-side configuration, please refer to their official [documentation page](#).

Communication chain

Overview

The overview of the communication chain is presented in the [setup overview](#).

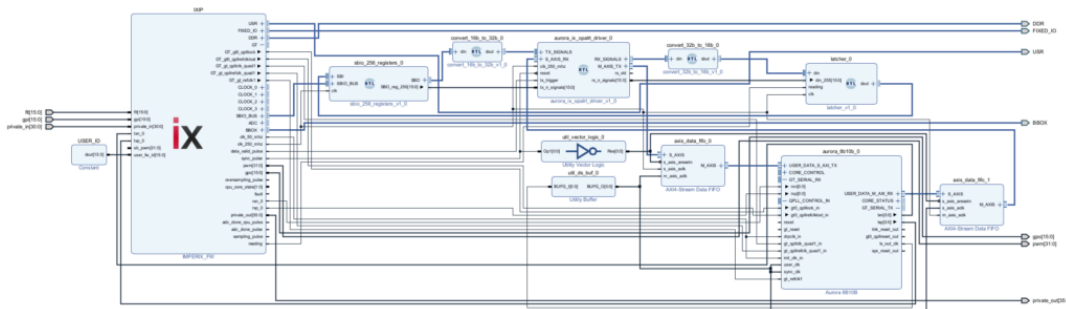


The main clock domain always runs at 250 MHz, while the frequency of the Aurora clock domain varies with the configuration of the Aurora IP. In this example, the frequency is 125 MHz with the configuration presented in the [Aurora parameters](#) section.

Vivado project

The Vivado project is provided in the form of generation scripts. As explained in the [PN109](#), the scripts automatically create and open the project illustrated below. The bitstream can be directly generated by simply pressing **Generate Bitstream** in the left navigation bar in the Vivado environment.

As provided in this example source files, the driver supports the exchange of up to **50 signals** in each direction and the Aurora communication is linked to the **SFP 0 (UP)** port of the imperix controller.



Once generated, the bitstream can be loaded onto the imperix controller using [Cockpit](#). A reboot is required for the bitstream change to take effect.

To increase the number of exchanged signals or change the SFP port used for the Aurora communication, please refer to [How to exchange more signals](#) and [How to assign a different SFP port](#).

Modules/IPs description

The Vivado project contains the following VHDL modules and IPs.

Module name	Type	Description
sbio_256_registers	VHDL module	Instantiates and provides access to SBIO bus registers in the FPGA. More information is provided here .
convert_16b_to_32b	VHDL module	Converts the 16-bit words of the SBIO bus back into the 32-bit words of the payload.
aurora_ix_typhoon_driver	VHDL module	Custom driver provided by imperix to communicate with the Typhoon HIL simulators ; mainly acts as a parallel-to-serial transmitter and serial-to-parallel receiver.

convert_32b_to_16b	VHDL module	Converts the 32-bit words received from the simulator through the driver into 16-bit words compatible with the SBIO bus.
latcher	VHDL module	Ensures data coherency by preventing the update of the SBI registers while the CPU is reading.
AXI4-Stream Data FIFO	Vivado IP (Xilinx)	Handles the clock domain crossing between the main 250 MHz domain of the imperix firmware and the Aurora clock domain ; buffers the frame in the transmission direction.
Aurora 8B10B	Vivado IP (Xilinx)	Handles the Aurora communication and interfaces with the underlying hardware logic.
Utility Vector Logic	Vivado IP (Xilinx)	Converts the active-high reset signal from the sync_pulse into an active-low reset signal for the FIFOs.

Aurora parameters

The included Aurora IP comes with the correct configuration pre-applied for interfacing with Typhoon HIL simulators. When creating a project from scratch, the Aurora IP must be configured in Vivado with the specific parameters listed below.

Protocol	Aurora 8B10B	Interface	Framing
Line Width (Bytes)	4	Flow Control	None
Line Rate (Gbps)	5	Little Endian Support	No
Dataflow Mode	Duplex	CRC	No

Any unspecified settings should remain at their default values.

Experimental validation

Physical setup

The physical setup is straightforward:

1. Connect both devices to the network, so that they can be configured and monitored from the PC.

2. Connect the imperix controller to the Typhoon simulator with an SFP cable. As provided, this example considers the port **SFP 0 (UP)** on the controller and **SFP 1** on the simulator.
3. Turn on both devices.

Software-side setup

To experimentally validate the system:

1. Download the three software parts available in the [downloads](#) section.
2. Build and load the [Typhoon application](#) on the simulator.
3. [Generate the bitstream](#) for the imperix controller.
4. Load the bitstream on the imperix controller via Cockpit.
5. Build the [user application template](#) and launch it on the imperix controller via Cockpit.
6. Use Cockpit to monitor the exchanged signals.

The whole system should now be running.

Real-time monitoring

Connect to the imperix controller with Cockpit. Add a [scope](#) in the project (from the **Modules tab** in the top bar) and drag-and-drop the variables of interest.

The exchanged signals can now be monitored in real-time in Cockpit. As expected, the amplitude of the transmitted signals is multiplied by two in the simulator.



The exchanged signals can also be monitored from Typhoon HIL SCADA, as described in the [Typhoon HIL application](#) section.

