PLEORA TECHNOLOGIES INC.

# iPORT NTx-Ten Embedded Video Interface
## User Guide

Pleora Technologies

## Document Number

EX001-019-0001, Version 6.0, 11/16/19

# Table of Contents

# Chapter 1

## About this Guide

This chapter describes the purpose and scope of this guide and provides a list of complimentary guides.

The following topics are covered in this chapter:

# What this Guide Provides

This guide provides you with all of the information you need to connect the NTx-Ten Embedded Video Interface to your sensor and related electronics to create a camera or other imaging device. In this guide you will find a product overview, connector details, and mechanical drawings along with instructions for installing the Pleora eBUS™ SDK, establishing an Ethernet connection, and performing general configuration tasks to properly display video.

The last section of this guide provides Technical Support contact information for Pleora Technologies.

# Related Documents

The *iPORT NTx-Ten Embedded Video Interface User Guide* is complemented by the following guides:

- *eBUS Player Quick Start Guide*
- *eBUS Player User Guide*
- *eBUS SDK C++ API Quick Start Guide* and *eBUS SDK C++ API Help File*
- *eBUS SDK .NET API Quick Start Guide* and *eBUS SDK .NET API Help File*
- *eBUS SDK for Linux Quick Start Guide*
- *GigE Vision Standard*, version 2.0 available from the Automated Imaging Association (AIA) at www.visiononline.org
- *GenICam Standard Features Naming Convention*, available from the European Machine Vision Association (EMVA) at www.emva.org.

# Chapter 2

## About the iPORT NTx-Ten Embedded Video Interface

This chapter describes the embedded video interface, including the product variants and key features.

The following topics are covered in this chapter:

# The iPORT NTx-Ten Embedded Video Interface

Pleora's iPORT™ NTx-Ten Embedded Video Interface is a high-performance video transmitter that provides designers of industrial cameras and other imaging systems the ability to transmit images at over 8 Gbps. Built on industry-standard 10 GigE technology, the iPORT NTx-Ten Embedded Video Interface ensures fast time-to-market, compatibility with the GigE Vision® specification, and allows development teams to concentrate their efforts on their core competencies.

It is ideal for ultra-high-performance applications such as semiconductor, flat-panel, web, and other quality inspection systems, as well as medical and military imaging. The embedded video interface efficiently converts video data to IP packets, which are then sent with low, consistent latency over a 10 Gigabit Ethernet (10 GigE) link to receiving software. The embedded video interface is fully compatible with the GigE Vision and GenICam™ standards, ensuring seamless interoperation in multi-vendor deployments, as well as environments where GigE Vision is used over both 1 and 10 GigE links.

The iPORT NTx-Ten Embedded Video Interface transports the imaging data over industry-standard fiber-based links via an SFP+ (small formfactor pluggable) connector, and thus can be easily connected to off-the-shelf 10 GigE components, like network cards and switches.

The embedded video interface leverages Ethernet's flexible networking capabilities, such as multicasting. The flexibility afforded by the adoption of networked video allows for transmission over greater distances, the ability to use distributed computing methods, and the capability to locate analysis PCs safely away from inspection areas.

Signals from system elements, connected to the NTx-Ten Embedded Video Interface's GPIO (general purpose inputs and outputs) allow you to accurately synchronize and control the operation of conveyors, encoders, cameras, sorting mechanisms, and other components — either independently from or in conjunction with the host PC on the network.

Compatible with Pleora's feature-rich application toolkit, the eBUS™ SDK, the iPORT NTx-Ten Embedded Video Interface enables cost-effective development of network-based high-performance cameras and other imaging devices.

# Model Variants

The iPORT NTx-Ten Embedded Video Interface is supplied in these variants and is equipped with these parts, as listed in the following table.

Table 1: Model Variants

| **iPORT NTx-Ten Embedded Video Interface package variants** | |
|---|---|
| **iPORT NTx-Ten Embedded Video Interface OEM Board Set** | **Quantity** |
| iPORT NTx-Ten Embedded Video Interface Board Set | 1 |

| **iPORT NTx-Ten Embedded Video Interface Fiber Development Kit** | **Quantity** |
|---|---|
| iPORT NTx-Ten Embedded Video Interface Board Set (mounted on a heat sink) | 1 |
| 10 Gigabit Ethernet NIC | 1 |
| Fiber optic cable, 2 meters | 1 |
| SFP+ fiber modules | 2 |
| Power supply with 12 V power adapter | 1 |
| Pleora eBUS SDK, provided on USB stick (includes eBUS Player sample application) | 1 |

# Feature Set

The NTx-Ten Embedded Video Interface provides the features and functions available in the table below.

| **Hardware** | |
|---|---|
| User circuitry interface | Pixel bus and IO |
| FPGA | Arria® II GX<br>EP2AGX95F780 |
| Image buffer | 64-bits wide<br>256 MB DDR2 RAM |
| Persistent memory | 16-bit wide 8 MB parallel FLASH |
| **Inputs/Outputs** | |
| GPIO inputs | 4 x 1.8V LVCMOS |
| GPIO outputs | 2 x 1.8V LVCMOS |
| Camera control outputs | 8 x 2.5V LVCMOS |
| **Network** | |
| Gigabit Ethernet | 10 GigE (as per IEEE 802.3) |
| GigE Vision packet resend | Yes |
| Ethernet bandwidth | Over 8 Gbps |
| Unicast and multicast | Yes |
| DHCP | Yes |
| **Other** | |
| Power supply | 8-13V |
| Operating temperature | 0° to 45°C |
| Storage temperature | -40°C to 85°C |

| **Other (Cont'd)** | |
|---|---|
| Serial communication | 1 UART<br>2 configurable (UART, USRT) |
| PHY | AMCC QT2025 |
| Power connector | 4-pin |
| Ethernet connector | 10GBASE-SR, -LR, and –LRM using linear or limiting SFP+ modules |
| FPGA external static memory | 16-bit wide 2 MB PSRAM |
| Clock generator | Included |
| Boundary scan chain | FPGA and PHY |
| **Frame Grabber** | |
| Number of data channels | 1 or 2 |
| Video input | 3.3V LVTTL, 2.5V LVCMOS |
| Scan types | Area scan, and line scan |
| Monochrome | Yes |
| Supported pixel formats | Grayscale model<br>Bayer<br>RGB<br>YUV 4:2:2 |

*Continued on next page...*

| Frame Grabber, Cont'd | |
|---|---|
| Pixel depth (bits) | 8, 10, 12, 14, 16, 24, 30, 32, 36 |
| Pixel clock | Min: 20 MHz<br>Max: 125 MHz |
| Taps per data channel | Source one: 1, 2, 4, 8<br>Source two: 1, 2, 4 |
| Image width (pixels) | Min: 16<br>Default: 640<br>Max: 16,382<br>Increment: 2<br>Note: Values may change depending on the number of taps as well as the pixel depth in use. |
| Image height (pixels) | Min: 1<br>Default: 480<br>Max: 16,383<br>Increment: 1 |

| | |
|---|---|
| Windowing | Yes |
| Decimation | No |
| Tap Geometry | Available tap geometries are:<br>• Geometry_1X_1Y<br>• Geometry_1X2_1Y<br>• Geometry_1X<br>• Geometry_1X2<br>• Geometry_1X4_1Y<br>• Geometry_1X4<br>• Geometry_1X8_1Y<br>• Geometry_1X8<br>• Geometry_2X2E<br>Note: Other tap geometries require custom software-based reconstruction. |
| Data port mapping | No |
| Pixel shifting | No |
| Pixel inversion | No |
| Recording/playback | No |

# Selected GenICam Features

In addition to the mandatory GenICam features for any compliant GigE Vision device, the iPORT NTx-Ten Embedded Video Interface provides a number of additional features. Selected GenICam features are listed in the following table. The first four features in the table are mandatory GenICam features, which are present in every GigE Vision compatible device.

Table 2: Selected GenICam Features

| Feature | Description |
|---|---|
| Width* | Specifies the width of the image (in pixels). |
| Height* | Specifies the height of the image (in pixels). |
| OffsetX | Specifies the horizontal image offset (in pixels). |
| OffsetY | Specifies the vertical image offset (in pixels). |
| PixelFormat* | Specifies the format of the pixel provided by the device. Available pixel formats are:<br>• Monochrome pixel formats, 8 to 16 bits<br>• Bayer pixel formats, 8 to 16 bits<br>• RGB pixel formats, 8 to 12 bits per component<br>• RGBA, 8-bit |
| DeviceReset | Resets the Embedded Video Interface to its power up state. |
| SourceCount* | Controls the number of sources supported by the device. |
| DeviceScanType* | Specifies the sensor scan type, such as areascan or linescan. |
| DeviceTapGeometry* | Describes the geometrical properties characterizing the taps of a Camera Link camera as seen from the frame grabber or acquisition card. This device tap geometry feature is defined in the GenICam SFNC.<br>Available tap geometries are:<br>• Geometry_1X_1Y<br>• Geometry_1X2_1Y<br>• Geometry_1X<br>• Geometry_1X2<br>• Geometry_1X4_1Y<br>• Geometry_1X4<br>• Geometry_1X8_1Y<br>• Geometry_1X8<br>• Geometry_2X2E<br>Note: Other tap geometries require custom software-based reconstruction. |
| SensorDigitizationTaps* | Specifies the number of digitized samples that are simultaneously output by the camera A/D conversion stage. |
| DigitizedImageWidth* | Width of the image provided by the camera (in pixels). |
| DigitizedImageHeight* | Height of the image provided by the camera (in pixels). |

* These features are interrelated. When you change any of these values, the embedded video interface may automatically adjust the other values to ensure the configuration is valid.

# Chapter 3

## iPORT NTx-Ten Embedded Video Interface Connections

This chapter describes the NTx-Ten Embedded Video Interface connections. It also includes pinouts for the user circuitry and power connectors, Camera Link signal mapping, and FPGA selection pins.

The following topics are covered in this chapter:

# Connector Locations

The following figure and table provide details about the NTx-Ten Embedded Video Interface connectors.

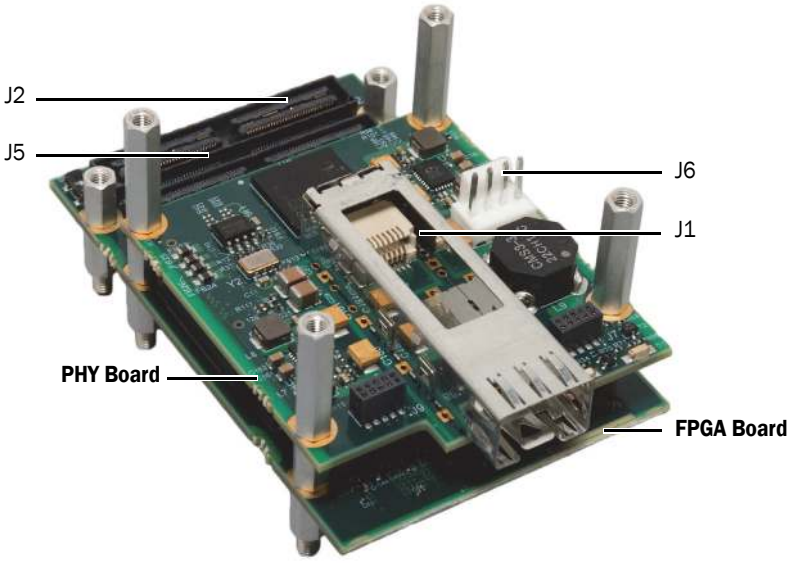



Table 3: Embedded Video Interface Connectors

| ID | Location | Type | Description |
|---|---|---|---|
| J1 | PHY Board | SFP+ Ethernet connector | Interfaces the embedded video interface to Ethernet networks, as specified in IEEE 802.3.<br>The Ethernet interface operates at 10 gigabits per second (Gbps), and supports Internet Protocol Version 4 (IPv4). |
| J6 | PHY Board | 4-pin power connector | Receives 8-13V of unfiltered DC input.<br>The power consumption of the embedded video interface is approximately 9 W (dependent on the SFP+ module in use). |
| J2, J5 | FPGA board | 120-pin user circuitry connector | Interfaces directly to the camera head or other external device. The user circuitry connectors allow you to integrate your NTx-Ten embedded video interface and camera head into a compact, single unit without the need for bulky video connectors. |

# Power Connector Pinouts

Receives 8-13V of unfiltered DC input. The power consumption of the embedded video interface is approximately 9 W (depending on the SFP+ module in use).

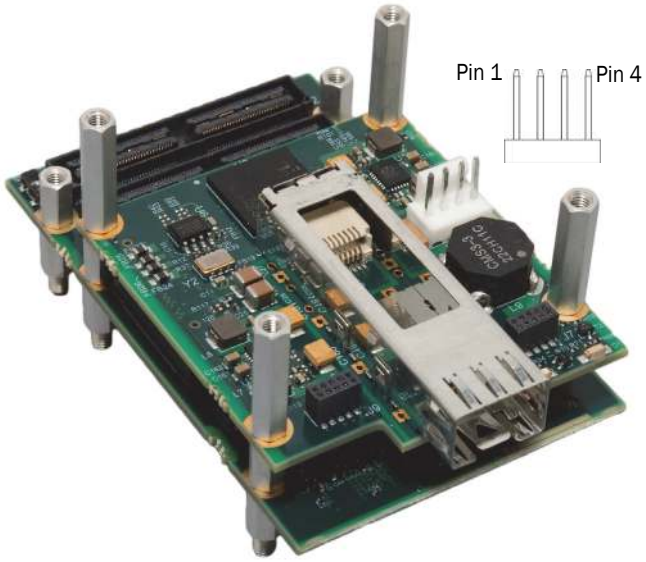



Table 4: Power Connector Pinouts

| Pin | Signal name |
|---|---|
| 1 | RET |
| 2 | PWR |
| 3 | PWR |
| 4 | RET |

# Support for SFP+ Modules

The NTx-Ten Embedded Video Interface Ethernet PHY supports SFP+ modules for fiber interfaces. Fiber Optic SFP+ modules are active and drive short and long-range cable.

In environments where ambient temperatures are high, we recommend that you use a heat-sink on the SFP+ cage. Additionally, industrial temperature SFP+ modules are available to extend the temperature of the environment where the NTx-Ten Embedded Video Interface can safely operate.

# User Circuitry Connectors

The NTx-Ten Embedded Video Interface contains two 120-pin user circuitry connectors, J2 and J5, that interface directly to the camera head or other external device. Each user circuitry connector contains specialized pinouts that allow you to integrate your NTx-Ten Embedded Video Interface and camera head into a compact, single unit without the need for bulky video connectors.

The GPIO in J2 allow the NTx-Ten Embedded Video Interface to control external machinery, or be controlled by external circuitry. The embedded video interface's GPIO are versatile and programmable, and the potential uses depend on your application.

The J5 connector has these features and functions:

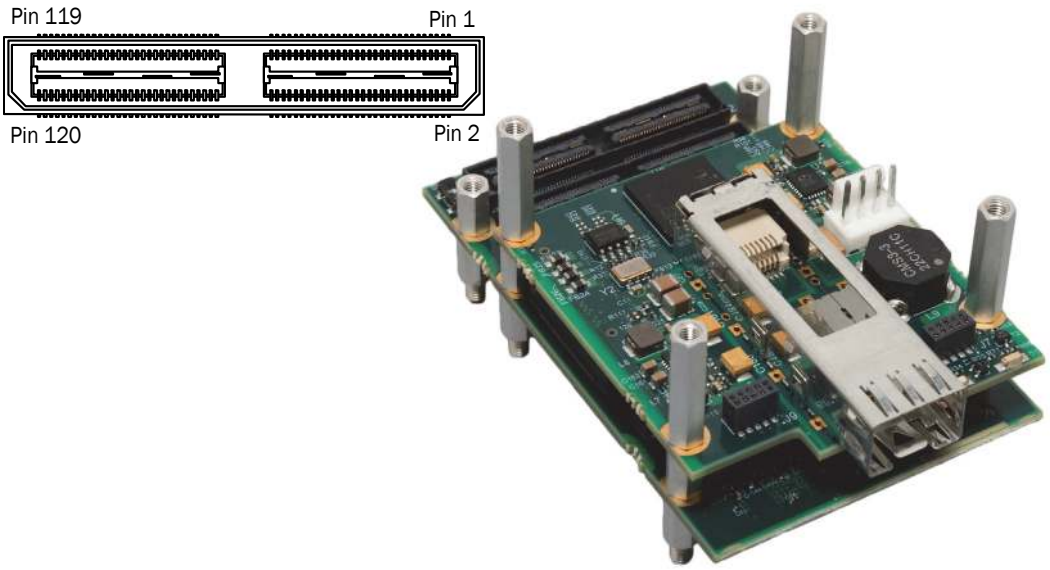
- 3.3V and 2.5V power supplies
- 2.5 V pixel bus interface
- 1 FPGA load selection input signal
- Power-on-reset (POR)
- Serial communication interfaces
- Ground pins
- Vin filtered power pins

The J2 connector has these features and functions:

- 3.3V, 2.5V, and 1.8V power supplies
- 1.8V GPIO
- 2.5 V pixel bus interface
- 33.333 MHz system clock, output by the NTx-Ten Embedded Video Interface
- Serial communication interfaces
- Ground pins
- $V_{in}$ filtered power pins

## User Circuitry Pinout Orientation

The pinout orientation of the Samtec QSH 120-pin vertical connector is illustrated in the following figure.



## Dual and Single-Source Mode

The NTx-Ten Embedded Video Interface can be used to receive video from two video sources. Or, it can be used in single-source mode.

### Dual Source Mode (48-Bit Pixel Bus Data for Each Video Source)

To receive video from two video sources, set the **SourceCount** feature to 2. Table 6 and Table 7 provide the user circuitry pinout descriptions for the dual source configuration.

### Single-Source Mode (64 or 96-Bit Pixel Bus Data for a Single Video Source)

By concatenating the source #1 and source #2 pixel data signals on the J5 and J2 connectors, you can make 64-bit or 96-bit pixel bus data available for one sensor. This is useful when you want to transmit video from a single sensor that exposes more than 4 taps.

Table 5: Single-Source Mode — 64 and 96-Bit Pixel Bus Data

| Configuration | Details |
|---|---|
| 64-bit pixel bus data | • Set the **SourceCount** feature to **1**.<br>• Available only with the **1X8_1Y** device tap geometry.<br>• Available only with 8-bit pixel formats, such as Mono8.<br>• Concatenate the **PB0_DATA[47:0]** (source #1) and **PB1_DATA[15:0]** (source #2) pixel data pins.<br>• The most significant bit (MSB) is **PB1_DATA15** of source #2 and the least significant bit (LSB) is **PB0_DATA00** of source #1. |
| 96-bit pixel bus data | • Set the **SourceCount** feature to **1**.<br>• Available only with the **1X8_1Y** device tap geometry.<br>• Available only with 12-bit pixel formats, such as Mono12.<br>• Concatenate the source #1 and source #2 pixel data pins.<br>• The MSB is **PB1_DATA47** of source #2 and the LSB is **PB0_DATA00** of source #1. |

## User Circuitry Connector — J2 Connector Pinouts

The descriptions in Table 6 and Table 7 use terminology based on the Camera Link Standard.

Table 6: User Circuitry Connector (J2) Pinout Descriptions

| Pin | Type | Function |
|---|---|---|
| 1 | Power in /out | Filtered power pins to/from the NTx-Ten (VIN) |
| 2 | Power in /out | Filtered power pins to/from the NTx-Ten (VIN) |
| 3 | Ground | Ground |
| 4 | Ground | Ground |
| 5 | Input | Source #2 - Pixel Clock Y (PB1_CLKY) |
| 6 | Input | BULK 1 – UART and USRT Receive data |
| 7 | Input | Source #2 - Pixel Clock X (PB1_CLKX) |
| 8 | Input | Reserved |
| 9 | Ground | Ground |
| 10 | Ground | Ground |
| 11 | Input | Source #2 - Pixel Data 24 (PB1_DATA24) |
| 12 | Input | Source #2 – Pixel Data 25 (PB1_DATA25) |
| 13 | Input | Source #2 – Pixel Data 26 (PB1_DATA26) |
| 14 | Input | Source #2 – Pixel Data 27 (PB1_DATA27) |
| 15 | Ground | Ground |
| 16 | Ground | Ground |
| 17 | Input | Source #2 – Pixel Data 28 (PB1_DATA28) |
| 18 | Input | Source #2 – Pixel Data 31 (PB1_DATA31) |
| 19 | Input | Source #2 – Pixel Data 29 (PB1_DATA29) |
| 20 | Input | Source #2 – Pixel Data 32 (PB1_DATA32) |
| 21 | Ground | Ground |
| 22 | Ground | Ground |
| 23 | Input | Source #2 – Pixel Data 33 (PB1_DATA33) |
| 24 | Input | Source #2 – Pixel Data 34 (PB1_DATA34) |
| 25 | Input | Source #2 – Pixel Data 38 (PB1_DATA38) |
| 26 | Input | Source #2 – Pixel Data 39 (PB1_DATA39) |
| 27 | Ground | Ground |
| 28 | Ground | Ground |
| 29 | Input | Source #2 – Pixel Data 35 (PB1_DATA35) |

Table 6: User Circuitry Connector (J2) Pinout Descriptions (Continued)

| Pin | Type | Function |
|---|---|---|
| 30 | Input | Source #2 – Pixel Data 36 (PB1_DATA36) |
| 31 | Input | Source #2 – Pixel Data 37 (PB1_DATA37) |
| 32 | Input | Source #2 – Pixel Data 40 (PB1_DATA40) |
| 33 | Ground | Ground |
| 34 | Ground | Ground |
| 35 | Input | Source #2 – Pixel Data 46 (PB1_DATA46) |
| 36 | Input | Source #2 – Pixel Data 47 (PB1_DATA47) |
| 37 | Input | Source #2 – Pixel Data 41 (PB1_DATA41) |
| 38 | Input | Source #2 – Pixel Data 42 (PB1_DATA42) |
| 39 | Ground | Ground |
| 40 | Ground | Ground |
| 41 | Input | Source #2 – Pixel Data 43 (PB1_DATA43) |
| 42 | Input | Source #2 – Pixel Data 44 (PB1_DATA44) |
| 43 | Input | Source #2 – Pixel Data 45 (PB1_DATA45) |
| 44 | Input | Source #2 – Spare Y (PB1_SPR_Y) |
| 45 | Ground | Ground |
| 46 | Ground | Ground |
| 47 | Input | Source #2 – Line Valid Y (PB1_LVAL_Y) |
| 48 | Input | Source #2 – Frame Valid Y (PB1_FVAL_Y) |
| 49 | Input | Source #2 – Data Valid Y (PB1_DVAL_Y) |
| 50 | Input | Source #2 – Pixel Data 30 (PB1_DATA30) |
| 51 | Ground | Ground |
| 52 | Ground | Ground |
| 53 | Output | Source #2 - Camera Control 1 (PB1_CC1) |
| 54 | Output | Source #2 - Camera Control 2 (PB1_CC2) |
| 55 | Output | Source #2 - Camera Control 3 (PB1_CC3) |
| 56 | Output | Source #2 - Camera Control 4 (PB1_CC4) |
| 57 | Ground | Ground |
| 58 | Ground | Ground |
| 59 | Output | 1.8 V 33.333MHz from the NTx-Ten Embedded Video Interface (system clock) |
| 60 | In/out | BULK 4 - USRT output clock when the interface is used in USRT mode. |

Table 6: User Circuitry Connector (J2) Pinout Descriptions (Continued)

| Pin | Type | Function |
|---|---|---|
| 61 | Input | Source #2 – Pixel Data 00 (PB1_DATA00) |
| 62 | Input | Source #2 – Pixel Data 01 (PB1_DATA01) |
| 63 | Input | Source #2 – Pixel Data 02 (PB1_DATA02) |
| 64 | Input | Source #2 – Pixel Data 03 (PB1_DATA03) |
| 65 | Input | Source #2 – Pixel Data 04 (PB1_DATA04) |
| 66 | Input | Source #2 – Pixel Data 07 (PB1_DATA07) |
| 67 | Input | Source #2 – Pixel Data 05 (PB1_DATA05) |
| 68 | Input | Source #2 – Pixel Data 08 (PB1_DATA08) |
| 69 | Input | Source #2 – Pixel Data 09 (PB1_DATA09) |
| 70 | Input | Source #2 – Pixel Data 10 (PB1_DATA10) |
| 71 | Input | Source #2 – Pixel Data 14 (PB1_DATA14) |
| 72 | Input | Source #2 – Pixel Data 15 (PB1_DATA15) |
| 73 | Input | Source #2 – Pixel Data 11 (PB1_DATA11) |
| 74 | Input | Source #2 – Pixel Data 12 (PB1_DATA12) |
| 75 | Input | Source #2 – Pixel Data 13 (PB1_DATA13) |
| 76 | Input | Source #2 – Pixel Data 16 (PB1_DATA16) |
| 77 | Power output | 3.3 V from the NTx-Ten Embedded Video Interface |
| 78 | Power output | 2.5 V from the NTx-Ten Embedded Video Interface |
| 79 | Input | Source #2 – Pixel Data 22 (PB1_DATA22) |
| 80 | Input | Source #2 – Pixel Data 23 (PB1_DATA23) |
| 81 | Input | Source #2 – Pixel Data 17 (PB1_DATA17) |
| 82 | Input | Source #2 – Pixel Data 18 (PB1_DATA18) |
| 83 | Input | Source #2 – Pixel Data 19 (PB1_DATA19) |
| 84 | Input | Source #2 – Pixel Data 20 (PB1_DATA20) |
| 85 | Input | Source #2 – Pixel Data 21 (PB1_DATA21) |
| 86 | Input | Source #2 – Spare X (PB1_SPR_X) |
| 87 | Input | Source #2 – Line Valid X (PB1_LVAL_X) |
| 88 | Input | Source #2 – Frame Valid X (PB1_FVAL_X) |
| 89 | Input | Source #2 – Data Valid X (PB1_DVAL_X) |
| 90 | Input | Source #2 – Pixel Data 06 (PB1_DATA06) |
| 91 | Output | Reserved |
| 92 | Output | Reserved |

Table 6: User Circuitry Connector (J2) Pinout Descriptions (Continued)

| Pin | Type | Function |
|---|---|---|
| 93 | Output | Reserved |
| 94 | Output | Reserved |
| 95 | Power output | 3.3 V from the NTx-Ten Embedded Video Interface |
| 96 | Power output | 2.5 V from the NTx-Ten Embedded Video Interface |
| 97 | Output | BULK 1 - UART and USRT transmit data |
| 98 | Output | Reserved |
| 99 | In/out | BULK 1 - USRT output clock when the interface is used in USRT mode. |
| 100 | In/out | Reserved |
| 101 | Output | Reserved |
| 102 | Output | Reserved |
| 103 | Input | GPIO input 0 |
| 104 | Input | GPIO input 1 |
| 105 | Input | Reserved |
| 106 | Input | Reserved |
| 107 | Power output | 1.8 V from the NTx-Ten Embedded Video Interface |
| 108 | Power output | 1.8 V from the NTx-Ten Embedded Video Interface |
| 109 | Output | GPIO output 0 |
| 110 | Output | GPIO output 1 |
| 111 | Output | Reserved |
| 112 | Output | Reserved |
| 113 | In/out | GPIO input 2 |
| 114 | In/out | GPIO input 3 |
| 115 | In/out | Reserved |
| 116 | In/out | Reserved |
| 117 | Output | Reserved |
| 118 | Output | Reserved |
| 119 | Output | Reserved |
| 120 | Output | Reserved |

For each input source, the sensor electronics must toggle all instances (X and Y) of Frame Valid, Line Valid, Data Valid, Spare, and Pixel Clock for that source at the same time. For example, pin 48 and 88 must toggle Frame Valid for source #2 at the same time.

## User Circuitry Connector — J5 Connector Pinouts

The user circuitry connector (J5) pinout descriptions are listed in the following table.

Table 7: User Circuitry Connector (J5) Pinout Descriptions

| Pin | Type | Function |
|---|---|---|
| 1 | Power in/power out | Filtered power pins to/from the NTx-Ten Embedded Video Interface (VIN) |
| 2 | Power in/power out | Filtered power pins to/from the NTx-Ten Embedded Video Interface (VIN) |
| 3 | Ground | Ground |
| 4 | Ground | Ground |
| 5 | Input | Source #1 – Pixel Clock X (PB0_CLKX) |
| 6 | Input | Source #1 – Pixel Clock Y (PB0_CLKY) |
| 7 | Input | BULK 0 - UART Receive data |
| 8 | Input | Reserved |
| 9 | Ground | Ground |
| 10 | Ground | Ground |
| 11 | Input | Source #1 – Pixel Data 24 (PB0_DATA24) |
| 12 | Input | Source #1 – Pixel Data 25 (PB0_DATA25) |
| 13 | Input | Source #1 – Pixel Data 26 (PB0_DATA26) |
| 14 | Input | Source #1 – Pixel Data 27 (PB0_DATA27) |
| 15 | Ground | Ground |
| 16 | Ground | Ground |
| 17 | Input | Source #1 – Pixel Data 28 (PB0_DATA28) |
| 18 | Input | Source #1 – Pixel Data 31 (PB0_DATA31) |
| 19 | Input | Source #1 – Pixel Data 29 (PB0_DATA29) |
| 20 | Input | Source #1 – Pixel Data 32 (PB0_DATA32) |
| 21 | Ground | Ground |
| 22 | Ground | Ground |
| 23 | Input | Source #1 – Pixel Data 33 (PB0_DATA33) |
| 24 | Input | Source #1 – Pixel Data 34 (PB0_DATA34) |
| 25 | Input | Source #1 – Pixel Data 38 (PB0_DATA38) |
| 26 | Input | Source #1 – Pixel Data 39 (PB0_DATA39) |
| 27 | Ground | Ground |
| 28 | Ground | Ground |
| 29 | Input | Source #1 – Pixel Data 35 (PB0_DATA35) |

Table 7: User Circuitry Connector (J5) Pinout Descriptions (Continued)

| Pin | Type | Function |
|---|---|---|
| 30 | Input | Source #1 – Pixel Data 36 (PB0_DATA36) |
| 31 | Input | Source #1 – Pixel Data 37 (PB0_DATA37) |
| 32 | Input | Source #1 – Pixel Data 40 (PB0_DATA40) |
| 33 | Ground | Ground |
| 34 | Ground | Ground |
| 35 | Input | Source #1 – Pixel Data 46 (PB0_DATA46) |
| 36 | Input | Source #1 – Pixel Data 47 (PB0_DATA47) |
| 37 | Input | Source #1 – Pixel Data 41 (PB0_DATA41) |
| 38 | Input | Source #1 – Pixel Data 42 (PB0_DATA42) |
| 39 | Ground | Ground |
| 40 | Ground | Ground |
| 41 | Input | Source #1 – Pixel Data 43 (PB0_DATA43) |
| 42 | Input | Source #1 – Pixel Data 44 (PB0_DATA44) |
| 43 | Input | Source #1 – Pixel Data 45 (PB0_DATA45) |
| 44 | Input | Source #1 – Spare Y (PB0_SPR_Y) |
| 45 | Ground | Ground |
| 46 | Ground | Ground |
| 47 | Input | Source #1 – Line Valid Y (PB0_LVAL_Y) |
| 48 | Input | Source #1 – Frame Valid Y (PB0_FVAL_Y)) |
| 49 | Input | Source #1 – Data Valid Y (PB0_DVAL_Y) |
| 50 | Input | Source #1 – Pixel Data 30 (PB0_DATA30) |
| 51 | Ground | Ground |
| 52 | Ground | Ground |
| 53 | Output | Source #1 - Camera Control 1 (PB0_CC1) |
| 54 | Output | Source #1 - Camera Control 2 (PB0_CC2) |
| 55 | Output | Source #1 - Camera Control 3 (PB0_CC3) |
| 56 | Output | Source #1 - Camera Control 4 (PB0_CC4) |
| 57 | Ground | Ground |
| 58 | Ground | Ground |
| 59 | Input | FPGA load selection |
| 60 | Open drain, input/output | Power-on reset from/to the NTx-Ten Embedded Video Interface. This is an active low, open drain signal pulled up to 2.5V on the NTx-Ten Embedded Video Interface (PWR_ON_RST#). |

Table 7: User Circuitry Connector (J5) Pinout Descriptions (Continued)

| Pin | Type | Function |
|---|---|---|
| 61 | Input | Source #1 – Pixel Data 00 (PB0_DATA00) |
| 62 | Input | Source #1 – Pixel Data 01 (PB0_DATA01) |
| 63 | Input | Source #1 – Pixel Data 02 (PB0_DATA02) |
| 64 | Input | Source #1 – Pixel Data 03 (PB0_DATA03) |
| 65 | Input | Source #1 – Pixel Data 04 (PB0_DATA04) |
| 66 | Input | Source #1 – Pixel Data 07 (PB0_DATA07) |
| 67 | Input | Source #1 – Pixel Data 05 (PB0_DATA05) |
| 68 | Input | Source #1 – Pixel Data 08 (PB0_DATA08) |
| 69 | Input | Source #1 – Pixel Data 09 (PB0_DATA09) |
| 70 | Input | Source #1 – Pixel Data 10 (PB0_DATA10) |
| 71 | Input | Source #1 – Pixel Data 14 (PB0_DATA14) |
| 72 | Input | Source #1 – Pixel Data 15 (PB0_DATA15) |
| 73 | Input | Source #1 – Pixel Data 11 (PB0_DATA11) |
| 74 | Input | Source #1 – Pixel Data 12 (PB0_DATA12) |
| 75 | Input | Source #1 – Pixel Data 13 (PB0_DATA13) |
| 76 | Input | Source #1 – Pixel Data 16 (PB0_DATA16) |
| 77 | Power output | 3.3 V from the NTx-Ten Embedded Video Interface |
| 78 | Power output | 2.5 V from the NTx-Ten Embedded Video Interface |
| 79 | Input | Source #1 – Pixel Data 22 (PB0_DATA22) |
| 80 | Input | Source #1 – Pixel Data 23 (PB0_DATA23) |
| 81 | Input | Source #1 – Pixel Data 17 (PB0_DATA17) |
| 82 | Input | Source #1 – Pixel Data 18 (PB0_DATA18) |
| 83 | Input | Source #1 – Pixel Data 19 (PB0_DATA19) |
| 84 | Input | Source #1 – Pixel Data 20 (PB0_DATA20) |
| 85 | Input | Source #1 – Pixel Data 21 (PB0_DATA21) |
| 86 | Input | Source #1 – Spare X (PB0_SPR_X) |
| 87 | Input | Source #1 – Line Valid X (PB0_LVAL_X) |
| 88 | Input | Source #1 – Frame Valid X (PB0_FVAL_X) |
| 89 | Input | Source #1 – Data Valid X (PB0_DVAL_X) |
| 90 | Input | Source #1 – Pixel Data 06 (PB0_DATA06) |
| 91 | Input | Reserved |
| 92 | Input | Reserved |

Table 7: User Circuitry Connector (J5) Pinout Descriptions (Continued)

| Pin | Type | Function |
|---|---|---|
| 93 | Input | Reserved |
| 94 | Input | Reserved |
| 95 | Power output | 3.3 V from the NTx Ten Embedded Video Interface |
| 96 | Power output | 2.5 V from the NTx Ten Embedded Video Interface |
| 97 | Output | BULK 0 - UART transmit data |
| 98 | Output | Reserved |
| 99 | Output | BULK 4 - UART and USRT transmit data |
| 100 | Input | BULK 4 - UART and USRT receive data |
| 101 | Input | Reserved |
| 102 | Input | Reserved |
| 103 | Output | Reserved |
| 104 | Output | Reserved |
| 105 | Output | Reserved |
| 106 | Input | Reserved |
| 107 | Power output | 3.3 V from the NTx-Ten Embedded Video Interface |
| 108 | Power output | 2.5 V from the NTx-Ten Embedded Video Interface |
| 109 | Input | Reserved |
| 110 | Output | Reserved |
| 111 | Output | Reserved |
| 112 | Output | Reserved |
| 113 | Input | Reserved |
| 114 | Input | Reserved |
| 115 | Output | Reserved |
| 116 | Output | Reserved |
| 117 | Output | Reserved |
| 118 | Input | Reserved |
| 119 | Input | Reserved |
| 120 | Output | Reserved |

For each source, the sensor electronics must toggle all instances (X and Y) of Frame Valid, Line Valid, Data Valid, Spare, and Pixel Clock for that source, at the same time. For example, pin 48 and 88 must toggle Frame Valid for source #2 at the same time.

# Chapter 4

## Status LEDs

The status LEDs indicate the operating status of the NTx-Ten Embedded Video Interface's network connection and firmware. The following figure and table describe the status LEDs.

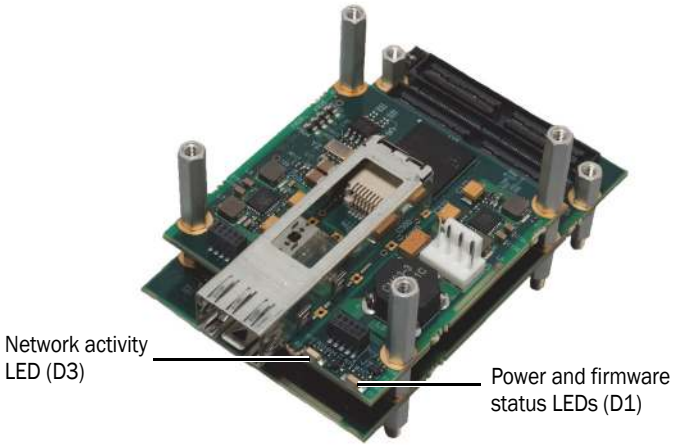


Table 8: Status LEDs

| LED | ID | Description |
|---|---|---|
| Link | D3 | The green LED provides the status of the network connection as well as the status of packets received from the network.<br>• If this LED is on and not flashing, the network is connected at 10Gbps.<br>• If the LED is off and not flashing, the network is not connected.<br>• If the LED is flashing, the network is connected at 10Gbps and Ethernet packets are being received.<br>The orange LED provides the status of packets transmitted to the network.<br>• If the LED is off and not flashing, Ethernet packets are not being transmitted.<br>• If the LED is flashing, Ethernet packets are being transmitted. |
| Power/FPGA | D1 | The green LED indicates whether or not the embedded video interface is receiving power.<br>• If this LED is on and not flashing, the embedded video interface is receiving power.<br>• If this LED is off and not flashing, the embedded video interface is not receiving power.<br>Note: This LED can also be off and not flashing for two or three seconds while the FPGA is being configured. If both FPGA loads are corrupted, this LED can remain off even if the embedded video interface is receiving power.<br>The orange LED indicates that the embedded video interface firmware loads are corrupted. Contact Pleora support. |

# Chapter 5

## Pixel Bus Timing

This chapter describes the interface that is responsible for transmitting data from the camera to the embedded video interface.

The following topics are covered in this chapter:

# Pixel Bus Timing Overview

The NTx-Ten Embedded Video Interface Pixel Bus transmits data from the camera to the embedded video interface in a format similar to deserialized Camera Link Standard data, as shown in the following image.

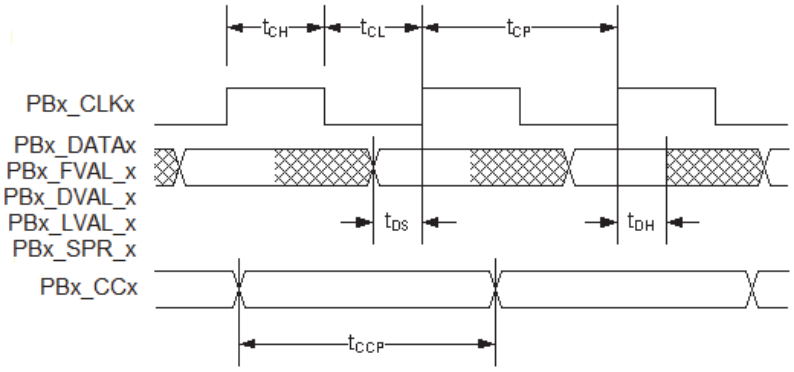



Table 9: Sub-clock Delays on the Camera Interface

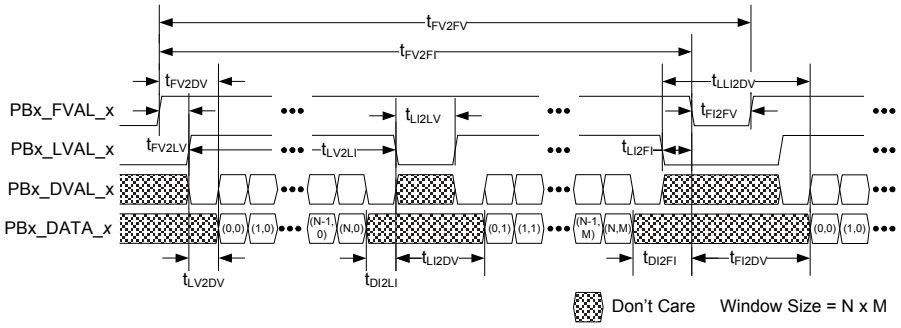
| Parameter | Symbol | Minimum | Maximum | Notes |
|---|---|---|---|---|
| PBx_CLKx high-level width | $t_{CH}$ | 3.2 ns | N/A | N/A |
| PBx_CLKx low-level width | $t_{CL}$ | 3.2 ns | N/A | N/A |
| PBx_CLKx frequency | $f_{CP}$ | 20 MHz | 125 MHz[a] | N/A |
| PBx_CLKx clock period | $t_{CP}$ | 8.0 ns | N/A | N/A |
| PBx_DATAx setup time | $t_{DS}$ | 2 ns | N/A | By design |
| PBx_DATAx hold time | $t_{DH}$ | 2 ns | N/A | By design |
| PBx_CCx pulse width | $t_{CCP}$ | 30 ns | N/A | Asynchronous with respect to PBx_CLK. |

a.To ensure optimal performance, ensure that the output data rate does not exceed 8.3 Gbps.

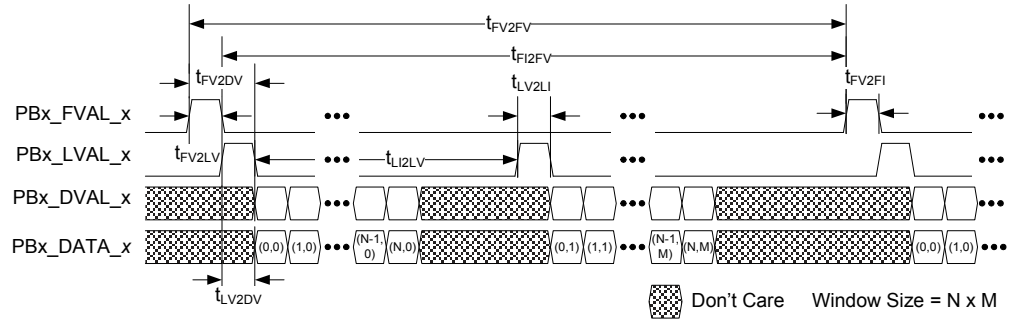
# Pixel Bus Signals

The output of the camera must match the format of the NTx-Ten Embedded Video Interface. You should select a case for your application and then refer to, "Timing Values for All Cases" on page 28. The stated timing restrictions are minimum values.

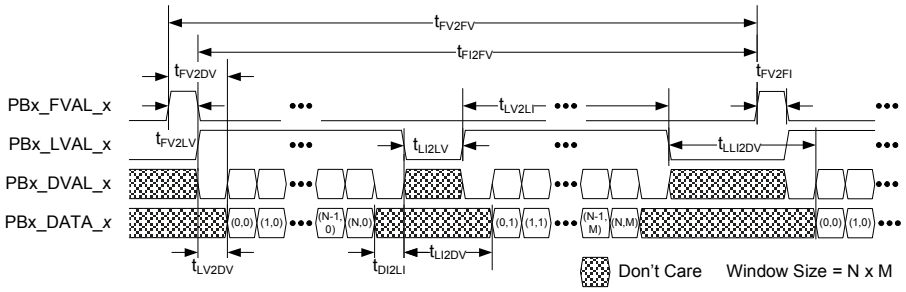
## Case 1: FVAL and LVAL are Level-Sensitive




## Case 2: FVAL and LVAL are Edge-Sensitive




## Case 3: FVAL is Edge-Sensitive and LVAL is Level-Sensitive



## Timing Values for All Cases

The timing values stated in the following table are minimum values only.

Table 10: Timing Values for All Cases

| From | To | Symbol | Case 1 (level) ($t_{cp}$) | Case 2 (edge) ($t_{cp}$) | Case 3 (both) ($t_{cp}$) |
|---|---|---|---|---|---|
| FVAL valid | LVAL valid [a] | $t_{FV2LV}$ | 0 [b] | 0 | 1 |
| FVAL valid | Data valid [a,c,d] | $t_{FV2DV}$ | 0 [b] | 16 | 1 |
| LVAL valid | Data valid [a,c,d] | $t_{LV2DV}$ | 0 | 1 | 0 |
| LVAL valid | LVAL invalid [a] | $t_{LV2LI}$ | 1 | 1 | 1 |
| LVAL invalid | LVAL valid [a] | $t_{LI2LV}$ | 1 | 1 | 1 |
| LVAL invalid (Automatic Internal Re-trigger disabled) | Data valid [a,c,d] | $t_{LI2DV}$ | 1 | N/A | 1 |
| LVAL invalid (Automatic Internal Re-trigger enabled) | Data valid | $t_{LI2DV}$ | 16 | N/A | 16 |
| Data invalid | LVAL invalid [a,c,d] | $t_{DI2LI}$ | 0 | N/A | 0 |
| LVAL invalid | FVAL invalid [a] | $t_{LI2FI}$ | 0 [e] | N/A | N/A |
| Data invalid | FVAL invalid [a,c,d] | $t_{DI2FI}$ | 0 [e] | N/A | N/A |
| FVAL invalid | FVAL valid [a] | $t_{FI2FV}$ | 1 | 1 | 1 |
| FVAL invalid | Data valid [a,c,d] | $t_{FI2DV}$ | 1 | N/A | N/A |
| Last LVAL invalid | Data valid | $t_{LLI2DV}$ | 16 | N/A | 16 |
| FVAL valid | FVAL invalid | $t_{FV2FI}$ | 16 | 1 | 1 |
| FVAL valid | FVAL valid | $t_{2FV2FV}$ | 17 | 17 | 17 |

a. The valid state of FVAL and LVAL is high when they are set as level-high sensitive or rising-edge sensitive. Their valid state is low when they are set as level-low sensitive or falling-edge sensitive.

b. If LVAL is valid before FVAL becomes valid, the grabber drops the full line.

c. Data valid is defined by FVAL valid (note a), LVAL valid (note a), and DVAL valid (note e).

d. The valid state of DVAL is high when it is set as level-high sensitive, and low when set as level-low sensitive. DVAL is always valid in the grabber when the parameter PixelBusDataValidEnabled is off.

e. If FVAL becomes invalid and LVAL is still valid, the line is truncated.

# Chapter 6

## Pixel Bus Definitions

The Camera Link Standard defines the bit assignments for various pixel types. For your reference, the tables below list the Camera Link ports and the associated embedded video interface signal names.

PB0_DATAxx is source one. The same information applies to source two, which would be shown in the table as PB1_DATAxx.

Table 11: Camera Link Pixel Bus Definitions

| | Mono8/ Bayer_8 | | Mono10/ Bayer/ | | Mono12/ Bayer/12 | | Mono14/ Bayer/14 | | Mono16/ Bayer/16 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Tap | Bit | Tap | Bit | Tap | Bit | Tap | Bit | Tap | Bit |
| PB0_DATA00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PB0_DATA01 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| PB0_DATA02 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 |
| PB0_DATA03 | 0 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 0 | 3 |
| PB0_DATA04 | 0 | 4 | 0 | 4 | 0 | 4 | 0 | 4 | 0 | 4 |
| PB0_DATA05 | 0 | 5 | 0 | 5 | 0 | 5 | 0 | 5 | 0 | 5 |
| PB0_DATA06 | 0 | 6 | 0 | 6 | 0 | 6 | 0 | 6 | 0 | 6 |
| PB0_DATA07 | 0 | 7 | 0 | 7 | 0 | 7 | 0 | 7 | 0 | 7 |
| PB0_DATA08 | 1 | 0 | 0 | 8 | 0 | 8 | 1 | 8 | 1 | 8 |
| PB0_DATA09 | 1 | 1 | 0 | 9 | 0 | 9 | 1 | 9 | 1 | 9 |
| PB0_DATA10 | 1 | 2 | nc | nc | 0 | 10 | 1 | 10 | 1 | 10 |
| PB0_DATA11 | 1 | 3 | nc | nc | 0 | 11 | 1 | 11 | 1 | 11 |
| PB0_DATA12 | 1 | 4 | 1 | 8 | 1 | 8 | 1 | 12 | 1 | 12 |
| PB0_DATA13 | 1 | 5 | 1 | 9 | 1 | 9 | 1 | 13 | 1 | 13 |
| PB0_DATA14 | 1 | 6 | nc | nc | 1 | 10 | nc | nc | 1 | 14 |

Table 11: Camera Link Pixel Bus Definitions (Continued)

| | Mono8/ Bayer_8 | | Mono10/ Bayer/ | | Mono12/ Bayer/12 | | Mono14/ Bayer/14 | | Mono16/ Bayer/16 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Tap | Bit | Tap | Bit | Tap | Bit | Tap | Bit | Tap | Bit |
| PB0_DATA15 | 1 | 7 | nc | nc | 1 | 11 | nc | nc | 1 | 15 |
| PB0_DATA16 | 2 | 0 | 1 | 0 | 1 | 0 | nc | nc | nc | nc |
| PB0_DATA17 | 2 | 1 | 1 | 1 | 1 | 1 | nc | nc | nc | nc |
| PB0_DATA18 | 2 | 2 | 1 | 2 | 1 | 2 | nc | nc | nc | nc |
| PB0_DATA19 | 2 | 3 | 1 | 3 | 1 | 3 | nc | nc | nc | nc |
| PB0_DATA20 | 2 | 4 | 1 | 4 | 1 | 4 | nc | nc | nc | nc |
| PB0_DATA21 | 2 | 5 | 1 | 5 | 1 | 5 | nc | nc | nc | nc |
| PB0_DATA22 | 2 | 6 | 1 | 6 | 1 | 6 | nc | nc | nc | nc |
| PB0_DATA23 | 2 | 7 | 1 | 7 | 1 | 7 | nc | nc | nc | nc |
| PB0_DATA24 | 3 | 0 | 3 | 0 | 3 | 0 | nc | nc | nc | nc |
| PB0_DATA25 | 3 | 1 | 3 | 1 | 3 | 1 | nc | nc | nc | nc |
| PB0_DATA26 | 3 | 2 | 3 | 2 | 3 | 2 | nc | nc | nc | nc |
| PB0_DATA27 | 3 | 3 | 3 | 3 | 3 | 3 | nc | nc | nc | nc |
| PB0_DATA28 | 3 | 4 | 3 | 4 | 3 | 4 | nc | nc | nc | nc |
| PB0_DATA29 | 3 | 5 | 3 | 5 | 3 | 5 | nc | nc | nc | nc |
| PB0_DATA30 | 3 | 6 | 3 | 6 | 3 | 6 | nc | nc | nc | nc |
| PB0_DATA31 | 3 | 7 | 3 | 7 | 3 | 7 | nc | nc | nc | nc |
| PB0_DATA32 | nc | nc | 2 | 0 | 2 | 0 | nc | nc | nc | nc |
| PB0_DATA33 | nc | nc | 2 | 1 | 2 | 1 | nc | nc | nc | nc |
| PB0_DATA34 | nc | nc | 2 | 2 | 2 | 2 | nc | nc | nc | nc |
| PB0_DATA35 | nc | nc | 2 | 3 | 2 | 3 | nc | nc | nc | nc |
| PB0_DATA36 | nc | nc | 2 | 4 | 2 | 4 | nc | nc | nc | nc |
| PB0_DATA37 | nc | nc | 2 | 5 | 2 | 5 | nc | nc | nc | nc |
| PB0_DATA38 | nc | nc | 2 | 6 | 2 | 6 | nc | nc | nc | nc |
| PB0_DATA39 | nc | nc | 2 | 7 | 2 | 7 | nc | nc | nc | nc |
| PB0_DATA40 | nc | nc | 2 | 8 | 2 | 8 | nc | nc | nc | nc |
| PB0_DATA41 | nc | nc | 2 | 9 | 2 | 9 | nc | nc | nc | nc |
| PB0_DATA42 | nc | nc | nc | nc | 2 | 10 | nc | nc | nc | nc |
| PB0_DATA43 | nc | nc | nc | nc | 2 | 11 | nc | nc | nc | nc |
| PB0_DATA44 | nc | nc | 3 | 8 | 3 | 8 | nc | nc | nc | nc |
| PB0_DATA45 | nc | nc | 3 | 9 | 3 | 9 | nc | nc | nc | nc |

Table 11: Camera Link Pixel Bus Definitions (Continued)

| | Mono8/ Bayer_8 | | Mono10/ Bayer/ | | Mono12/ Bayer/12 | | Mono14/ Bayer/14 | | Mono16/ Bayer/16 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Tap | Bit | Tap | Bit | Tap | Bit | Tap | Bit | Tap | Bit |
| PB0_DATA46 | nc | nc | nc | nc | 3 | 10 | nc | nc | nc | nc |
| PB0_DATA47 | nc | nc | nc | nc | 3 | 11 | nc | nc | nc | nc |

Table 12: Camera Link Pixel Bus Definitions (Continued)

| | RGB8 | | RGB10 | | *RGBa8 | | RGB12 | |
|---|---|---|---|---|---|---|---|---|
| | Component | Bit | Component | Bit | Component | Bit | Component | Bit |
| PB0_DATA00 | R | 0 | R | 0 | R | 0 | R | 0 |
| PB0_DATA01 | R | 1 | R | 1 | R | 1 | R | 1 |
| PB0_DATA02 | R | 2 | R | 2 | R | 2 | R | 2 |
| PB0_DATA03 | R | 3 | R | 3 | R | 3 | R | 3 |
| PB0_DATA04 | R | 4 | R | 4 | R | 4 | R | 4 |
| PB0_DATA05 | R | 5 | R | 5 | R | 5 | R | 5 |
| PB0_DATA06 | R | 6 | R | 6 | R | 6 | R | 6 |
| PB0_DATA07 | R | 7 | R | 7 | R | 7 | R | 7 |
| PB0_DATA08 | G | 0 | R | 8 | G | 0 | R | 8 |
| PB0_DATA09 | G | 1 | R | 9 | G | 1 | R | 9 |
| PB0_DATA10 | G | 2 | nc | nc | G | 2 | R | 10 |
| PB0_DATA11 | G | 3 | nc | nc | G | 3 | R | 11 |
| PB0_DATA12 | G | 4 | B | 8 | G | 4 | B | 8 |
| PIXEL_DATA13 | G | 5 | B | 9 | G | 5 | B | 9 |
| PB0_DATA14 | G | 6 | B | nc | G | 6 | B | 10 |
| PB0_DATA15 | G | 7 | B | nc | G | 7 | B | 11 |
| PB0_DATA16 | B | 0 | B | 0 | B | 0 | B | 0 |
| PB0_DATA17 | B | 1 | B | 1 | B | 1 | B | 1 |
| PB0_DATA18 | B | 2 | B | 2 | B | 2 | B | 2 |
| PB0_DATA19 | B | 3 | B | 3 | B | 3 | B | 3 |
| PB0_DATA20 | B | 4 | B | 4 | B | 4 | B | 4 |
| PB0_DATA21 | B | 5 | B | 5 | B | 5 | B | 5 |
| PB0_DATA22 | B | 6 | B | 6 | B | 6 | B | 6 |
| PB0_DATA23 | B | 7 | B | 7 | B | 7 | B | 7 |
| PB0_DATA24 | nc | nc | nc | nc | A | 0 | nc | nc |
| PB0_DATA25 | nc | nc | nc | nc | A | 1 | nc | nc |
| PB0_DATA26 | nc | nc | nc | nc | A | 2 | nc | nc |

Table 12: Camera Link Pixel Bus Definitions (Continued)

| | RGB8 | | RGB10 | | *RGBa8 | | RGB12 | |
|---|---|---|---|---|---|---|---|---|
| | Component | Bit | Component | Bit | Component | Bit | Component | Bit |
| PB0_DATA27 | nc | nc | nc | nc | A | 3 | nc | nc |
| PB0_DATA28 | nc | nc | nc | nc | A | 4 | nc | nc |
| PB0_DATA29 | nc | nc | nc | nc | A | 5 | nc | nc |
| PB0_DATA30 | nc | nc | nc | nc | A | 6 | nc | nc |
| PB0_DATA31 | nc | nc | nc | nc | A | 7 | nc | nc |
| PB0_DATA32 | nc | nc | G | 0 | nc | nc | G | 0 |
| PB0_DATA33 | nc | nc | G | 1 | nc | nc | G | 1 |
| PB0_DATA34 | nc | nc | G | 2 | nc | nc | G | 2 |
| PB0_DATA35 | nc | nc | G | 3 | nc | nc | G | 3 |
| PB0_DATA36 | nc | nc | G | 4 | nc | nc | G | 4 |
| PB0_DATA37 | nc | nc | G | 5 | nc | nc | G | 5 |
| PB0_DATA38 | nc | nc | G | 6 | nc | nc | G | 6 |
| PB0_DATA39 | nc | nc | G | 7 | nc | nc | G | 7 |
| PB0_DATA40 | nc | nc | G | 8 | nc | nc | G | 8 |
| PB0_DATA41 | nc | nc | G | 9 | nc | nc | G | 9 |
| PB0_DATA42 | nc | nc | G | nc | nc | nc | G | 10 |
| PB0_DATA43 | nc | nc | G | nc | nc | nc | G | 11 |
| PB0_DATA44 | nc | nc | G | nc | nc | nc | nc | nc |
| PB0_DATA45 | nc | nc | G | nc | nc | nc | nc | nc |
| PB0_DATA46 | nc | nc | G | nc | nc | nc | nc | nc |
| PB0_DATA47 | nc | nc | G | nc | nc | nc | nc | nc |

* Not supported in the Camera Link Standard.

# Chapter 7

# NTx-Ten Embedded Video Interface Bulk Interfaces

This chapter describes the NTx-Ten Embedded Video Interface bulk interfaces and the supported protocols.

The following topics are covered in this chapter:

# Bulk Interfaces and Supported Protocols

The NTx-Ten Embedded Video Interface has three bulk interfaces: bulk 0, bulk1, and bulk 4. The three bulk interfaces support these protocols:

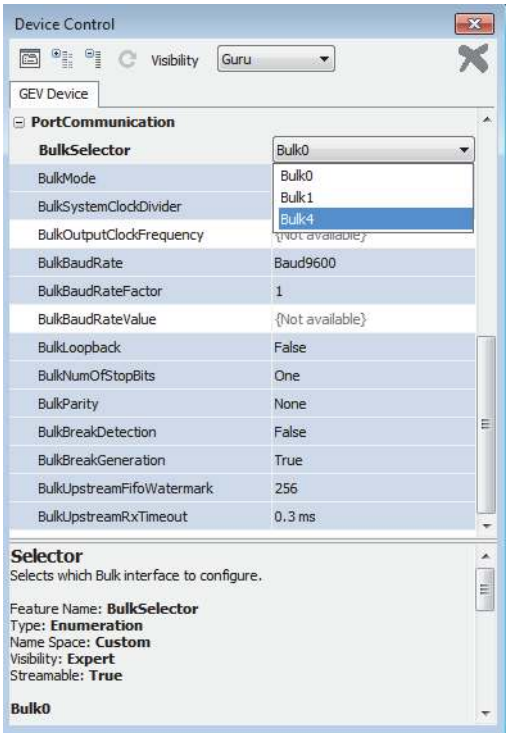
- Bulk 0: UART
- Bulk 1: UART, USRT
- Bulk 4: UART, USRT

The selection of the bulk interface protocols is controlled by an adjustable parameter in software at run-time.

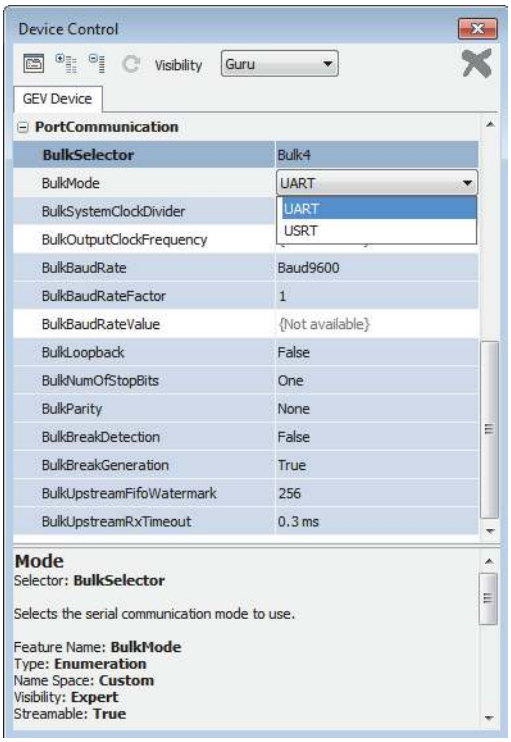
The signal and timing requirements for each protocol are explained in the sections that follow.

## To set the bulk interface

1. Start eBUS Player and connect to the embedded video interface.
   For more information, see "To start eBUS Player and connect to a device" on page 50.
2. If images are streaming, click the **Stop** button.
3. Under **Parameters and Controls**, click **Device control**.
4. Click **Expert or Guru** in the **Visibility** list.
5. Under **PortCommunication**, select a bulk interface from the **BulkSelector** list.



6. Under **PortCommunication**, select a protocol from the **BulkMode** list.




# UART Signals

The standard serial port communication uses the following signals:

- BULKn_TXD
- BULKn_RXD
- DGND (return)

# UART Timing

The NTx-Ten Embedded Video Interface supports these UART protocols:

- 8-bit data transfer
- 1 start bit
- Any parity (even, odd, or none)
- 1 or 2 stop bits

A number of preset baud rates can be used, as well as a more flexible baud rate factor, as shown in the following table.

Table 13: UART Baud Rates

| Baud factor, BF | Baud rate, BR [bps] | Notes |
|---|---|---|
| BF | 1/(BF*240 ns) | Programmable |
| 1 (min) | 4,166,667 | |
| 36 | 115,200 | Preset 6 |
| 72 | 57,600 | Preset 5 |
| 108 | 38,400 | Preset 4 |
| 144 | 28,800 | Preset 3 |
| 218 | 19,200 | Preset 2 |
| 290 | 14,400 | Preset 1 |
| 434 | 9,600 | Preset 0 (default) |
| 511 (max) | 8,154 | |

The following table provides the A.C. operating characteristics of the UART Interfaces.

Table 14: A.C Operating Characteristics of the UART Interfaces

| Parameter | Symbol | Min | Max | Units | Notes |
|---|---|---|---|---|---|
| Data Period | $^{t}$UART | 0.240 | 122.64 | µs | |
| Baud Rate | BR | 8,154 | 4,166,667 | bps | 1/$^{t}$UART |

# USRT Signals

The USRT (universal synchronous receiver/transmitter) serial interface resembles the UART interface, but adds a clock signal to enable synchronous communication.

Table 15: USRT Signal Nomenclature

| Embedded video interface signal | Generic signal |
|---|---|
| BULKx_RXD | RXD |
| BULKx_TXD | TXD |
| BULKx_SCK | SCK |

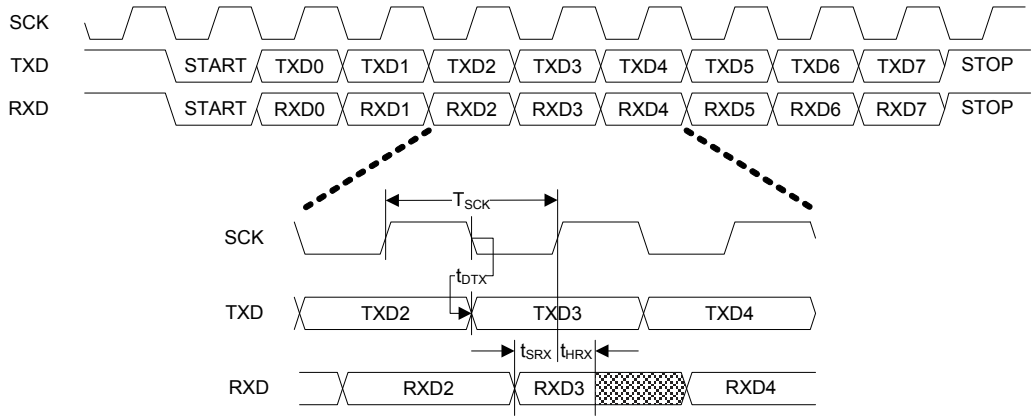
# USRT Timing

The following table lists the supported clock frequencies and periods for the USRT serial interface.

Table 16: Supported Clock Frequencies and Periods

| Clock divider | Clock period, $t_{SCK}$ (ns) | Clock frequency[a] (MHz) |
|---|---|---|
| By 2 | 60 | 16.667 |
| By 4 | 120 | 8.333 |
| By 8 | 240 | 4.167 |
| By 16 | 480 | 2.083 |
| By 32 | 960 | 1.042 |
| By 64 | 1920 | 0.521 |
| By 128 | 3840 | 0.260 |
| By 256 | 7680 | 0.130 |

a. To obtain the exact frequency, divide the 33.333 MHz clock speed by one of: 2, 4, 8, 16, 32, 64, 128, or 256.



# Chapter 8

## Thermal Requirements

This chapter provides information you need to properly cool the NTx-Ten Embedded Video Interface.

One way to cool the NTx-Ten Embedded Video Interface is by using a heat sink, which you can design.

The following topics are covered in this chapter

# NTx-Ten Embedded Video Interface Thermal Guidelines

The NTx-Ten Embedded Video Interface can consume more than 9 W of power, which can cause the Embedded Video Interface to become overheated, resulting in damaged embedded video interface components. If the NTx-Ten Embedded Video Interface is used with a properly designed heat sink, it can be safely deployed in a 65°C (or higher) environment.

The NTx-Ten Embedded Video Interface Development Kit includes a heat sink that provides enough surface area to adequately cool the NTx-Ten Embedded Video Interface FPGA and DDR2 memory as well as the PHY, which consumes a lot of power and must be cooled when operating in most enclosures.

# Designing your Own Heat Sink

You can design your own heat sink to use with the NTx-Ten Embedded Video Interface. The following table lists the optimal operating temperatures for key NTx-Ten Embedded Video Interface components.

Table 17: Optimal Operating Temperatures

| Component | Part number | Operating | Case | Junction | Theta-JC | Power | Max case |
|---|---|---|---|---|---|---|---|
| SFP+ | Example: Intel E10GSFPSR | 0-85°C | 0-70°C | | | | 70°C |
| FPGA | Altera EP2AGX95EF29C6N | | | 0-85°C | 0.1°C/W | 3.5 W | 84.75°C |
| PHY | Applied Micro Circuits Corporation (AMCC)<br>QT2025PRKD-1 | 0-80°C | | 110°C | 7.2°C/W | 1.6 W | 98.5°C |
| DDR2 | Samsung | -40-95°C | | 95°C | 6°C/W | 0.5 W | 92°C |
| PSRAM | Micron | -30-85°C | | | | | |
| Flash memory | Micron<br>PF48F3000P0ZBQEA | | -40-85°C | | | | |
| EPLD | Altera EPM240GF100C5N | 0-85°C | | 135°C | | | |
| Oscillator | Sunstu<br>**156MHz:**<br>SSMA348-156.250MHZ<br>**33MHz:**<br>SRZB48-33.333MHZ | -40-85°C | | 85°C | | | |
| Switcher | Texas Instruments<br>TPS73525DRVT | -40-85°C | | 85°C | | | |

Table 17: Optimal Operating Temperatures (Continued)

| Component | Part number | Operating | Case | Junction | Theta-JC | Power | Max case |
|---|---|---|---|---|---|---|---|
| Switcher | Linear Technology LTC3025EDC-1#PBF | -40-125 °C | | 125 °C | | | |
| Switcher | Linear Technology LTC3633EUFD#PBF | -40-125 °C | | 125 °C | | | |

# Chapter 9

# Installing the eBUS SDK

This chapter describes how to install the eBUS SDK, and also provides information about installing the required driver.

Before you can configure and control your embedded video interface, you must install the eBUS SDK.

The following topics are covered in this chapter:

# Installing the eBUS SDK

You can install the Pleora eBUS SDK on your computer to configure and control your embedded video interface. Consult the *eBUS Player Quick Start Guide* or *eBUS Player User Guide* for information about setting up and configuring your camera for connection to the embedded video interface.

The Pleora Technologies eBUS SDK contains an extensive library of sample applications, with source code, to create working applications for device configuration and control, image and data acquisition, and image display and diagnostics.

It is possible for you to configure the embedded video interface and GigE Vision compliant video sources using other GenICam compliant software, however, this guide provides you with the instructions you need to use the Pleora eBUS Player application.

# Installing the Driver and Configuring the NIC

Before you can configure the embedded video interface, use the Driver Installation Tool (included with the eBUS SDK) to install the correct driver. Then, set up your NIC.

## To install a Pleora driver

1. Click **Start** > **All Programs** > **eBUS** > **eBUS Driver Installation Tool**.
2. Under **GigE Vision**, click **Install**.

   After a moment the driver installs and the driver status changes to **Installed**. The driver is installed across all network adapters on your computer..


3. Close the eBUS Driver Installation Tool.

   You may be required to restart your computer.

To see the versions of the installed drivers, click **Help** > **About**.

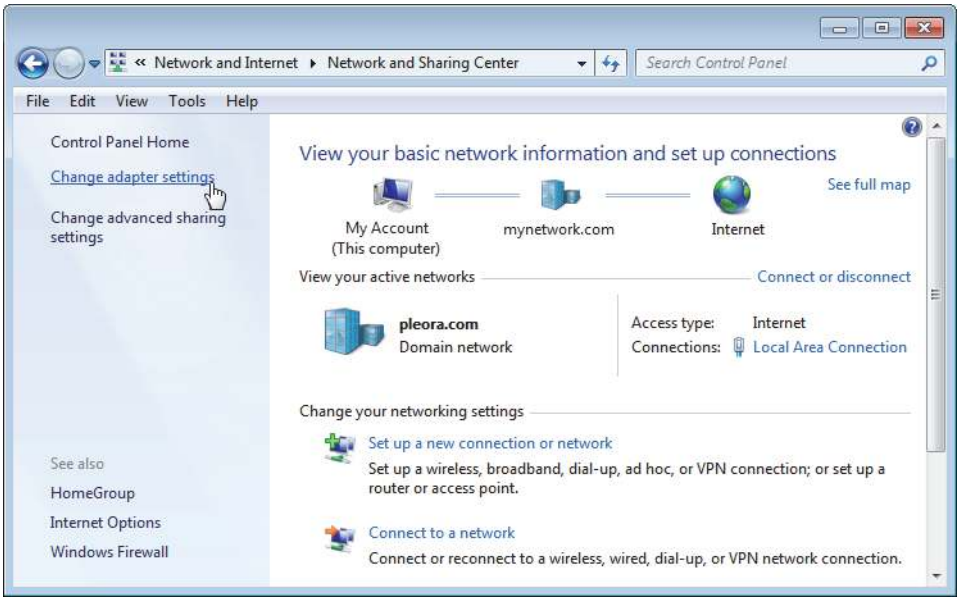
4. Select a driver that is suitable for your computer's NIC. If you require additional information to help choose the most appropriate driver, click **Learn more about drivers**.
5. Click **Do Nothing**.

### To configure an IP address for the NIC

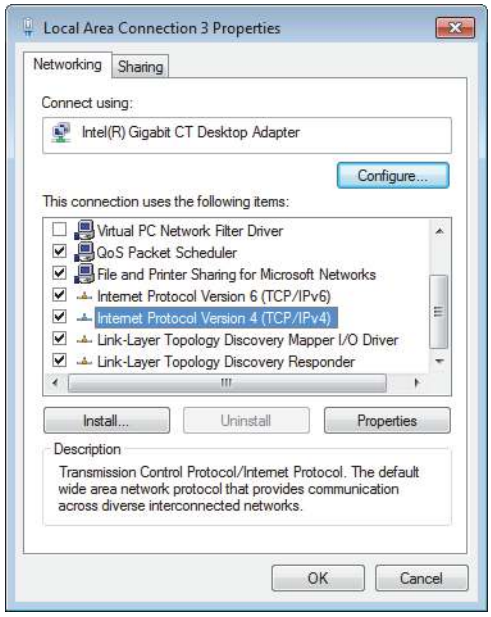
1. In the Windows Control Panel, click **Network and Internet**.

The instructions in this procedure are based on the Windows 7 operating system. The steps may vary depending on your computer's operating system.

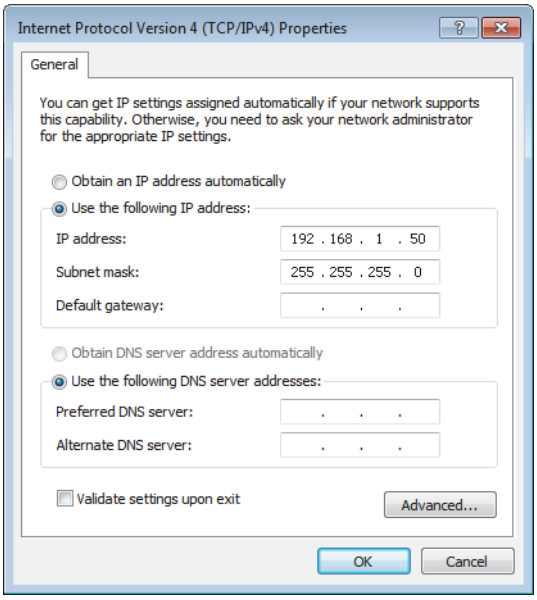

2. Click **Network and Sharing Center**.
3. In the left-hand panel, click **Change adapter settings**.



4. Right-click the NIC and then click **Properties**.
5. Click **Internet Protocol Version 4 (TCP/IPv4)** and then click **Properties**.




6. Select **Obtain an IP address automatically** or **Use the following IP address** to give the NIC an IP address.

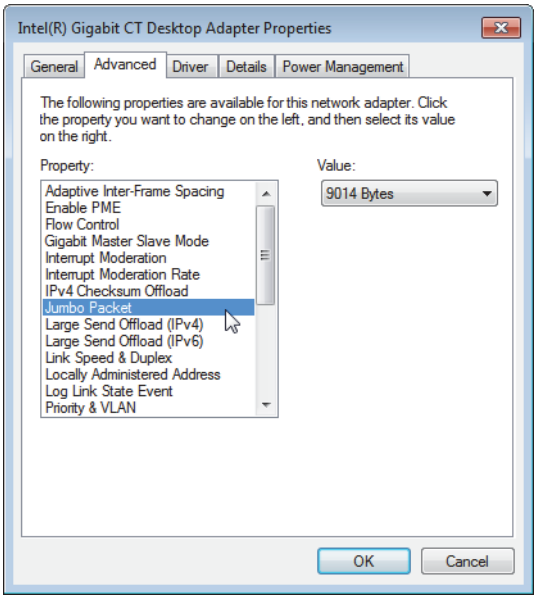



Static IP addressing allows for quicker detection and connection times on networks without a DHCP server, as the NIC does not have to negotiate a Link-Local Address (LLA).

7. Close the open dialog boxes to apply the changes and close the Control Panel.

8. Configure the NIC for jumbo packets (more often referred to as jumbo frames) and set the NIC's Rx Descriptor to the maximum available value. Using jumbo packets allows you to increase system performance. However, you must ensure your NIC and GigE switch (if applicable) support jumbo packets.

   To complete this task, right-click the NIC and click **Properties**. Then, click **Configure**. The exact configuration procedure, as well as the jumbo packet size limit, depends on the NIC.



# Chapter 10

## Connecting to the Embedded Video Interface and Configuring General Settings

After you have connected to the embedded video interface, you can provide it with a unique IP address on your network. When a connection is established, start eBUS Player and connect to the embedded video interface. Then, you can configure its image settings to ensure images are received and displayed properly. You can also configure the buffer options to reduce the likelihood of lost packets.

eBUS Player is documented in more detail in the *eBUS Player Quick Start Guide* and the *eBUS Player User Guide*. The *iPORT NTx-GigE Embedded Video Interface User Guide* provides you with the eBUS Player instructions and overviews required to set up and configure the embedded video interface.

The following topics are covered in this chapter:

# Connecting the Ethernet Cables and Confirming Image Streaming

The embedded video interface can communicate with your computer using either a direct connection or by connecting to a 10 GigE switch. This section explains how to connect the embedded video interface to a 10 GigE switch to confirm that images are streaming.

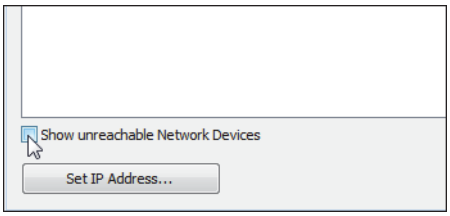
### To connect the Ethernet cables and apply power

1. Connect the embedded video interface to a 10 GigE port on your computer's NIC.
2. Apply power.

### To start eBUS Player and connect to a device

1. Start eBUS Player from the Windows **Start** menu.
2. Click **Select/Connect**.

   If the device does not appear in the list, click the **Show unreachable Network Devices** check box to show all devices.




3. In the **Device Selection** dialog box, click the embedded video interface.

If the IP address is not valid, a warning ( ) appears in the **Device Selection** dialog box. Provide the device with an IP address, as outlined in "Providing the Embedded Video Interface with an IP Address" on page 52.

4. Click **OK**.

   eBUS Player is now connected to the device.

### To confirm image streaming

1. Under **Acquisition Control**, click the source to which a camera is connected.
2. Click **Play** to stream live images.
3. After you confirm that images are streaming, click **Stop**.

If images do not stream, see the tips provided in "System Troubleshooting" on page 73.

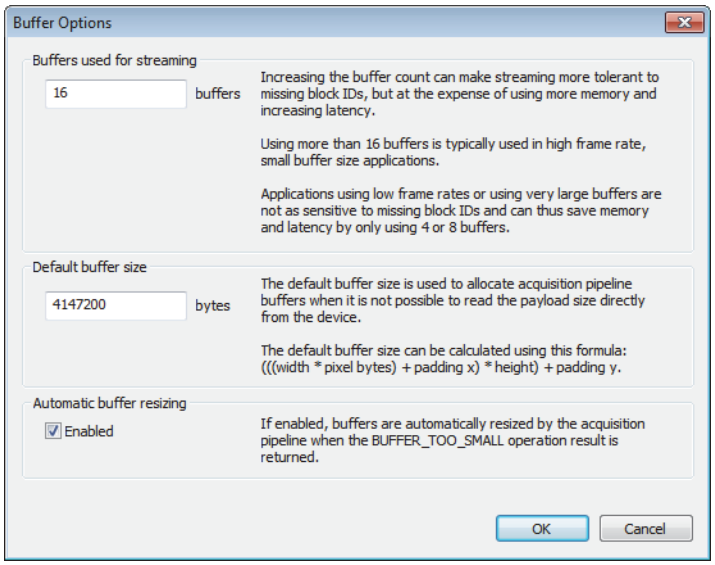
# Configuring the Buffers

You can increase the buffer count in eBUS Player to reduce the impact and likelihood of lost and out-of-order packets, and to make streaming more robust. A high number of buffers are needed in high frame rate applications, while a small number of buffers are needed for lower frame rates. Applications using a high number of buffers might experience greater latency.

## To configure the buffers

1. Start eBUS Player and connect to the embedded video interface.
   For more information, see "To start eBUS Player and connect to a device" on page 50.
2. Click **Tools** > **Buffer Options**.
3. Click the buffer option that suits your requirements.
4. Click **OK**.

Default size for streaming is 16 buffers.



# Providing the Embedded Video Interface with an IP Address

The embedded video interface requires an IP address to communicate on a video network. This address must be on the same subnet as the computer that is performing the configuration and receiving the image stream.

### To provide the embedded video interface with an IP address

1. Start eBUS Player.
2. Click **Select/Connect**.
3. Click the embedded video interface.
4. Click **Set IP Address**.
5. Provide the embedded video interface with a valid IP address and subnet mask. You can optionally provide a default gateway.

If you are using a unicast network configuration, the management entity/data receiver and the embedded video interface must be on the same subnet. The unicast network configuration is outlined in "Unicast Network Configuration" on page 62.

6. Click **OK** to close the **Set IP Address** dialog box.
7. Click **OK** to close the **Device Selection** dialog box.

## Configuring an Automatic/Persistent IP Address

The Device Control dialog box allows you to configure a persistent IP address for the embedded video interface. Alternatively, the embedded video interface can be configured to automatically obtain an IP address using Dynamic Host Configuration Protocol (DHCP) or Link Local Addressing (LLA). The embedded video interface uses its persistent IP address first, but if this option is set to **False**, it can be configured to attempt to obtain an address from a DHCP server. If this fails, it will use LLA to find an available IP address. LLA cannot be disabled and is always set to **True**.

### To configure a persistent IP address

1. Start eBUS Player and connect to the embedded video interface.
   For more information, see "To start eBUS Player and connect to a device" on page 50.
2. Under **Parameters and Controls**, click **Device control**.
3. Under **TransportLayerControl**, set the **GevCurrentIPConfigurationPersistentIP** feature to **True**.
4. Set the **GevPersistentIPAddress** feature to a valid IP address in the **GevPersistentIPAddress** field.
5. Set the **GevPersistentSubnetMask** feature to a valid subnet mask address.
6. Optionally, enter a valid default gateway in the **GevPersistentDefaultGateway** field.
7. Close the **Device Control** dialog box.
8. Power cycle the embedded video interface.

### To automatically configure an IP address

1. Start eBUS Player and connect to the embedded video interface.

   For more information, see "To start eBUS Player and connect to a device" on page 50.
2. Under **Parameters and Controls**, click **Device control**.
3. Under **TransportLayerControl**, set the **GevCurrentIPConfigurationPersistentIP** feature to **False**.
4. Set the **GevCurrentIPConfigurationLLA** and/or **GevCurrentIPConfigurationDHCP** values to **True**, depending on the type of automatic addressing you require.
5. Close the **Device Control** dialog box.
6. Power cycle the embedded video interface.

## Configuring the Embedded Video Interface's Image Settings

You can configure the embedded video interface's image settings, which provide the embedded video interface with information about the image coming from the camera. These settings allow the images to appear correctly.

The image settings are located under **ImageFormatControl** in the **Device Control** dialog box.

### To turn the test pattern on or off

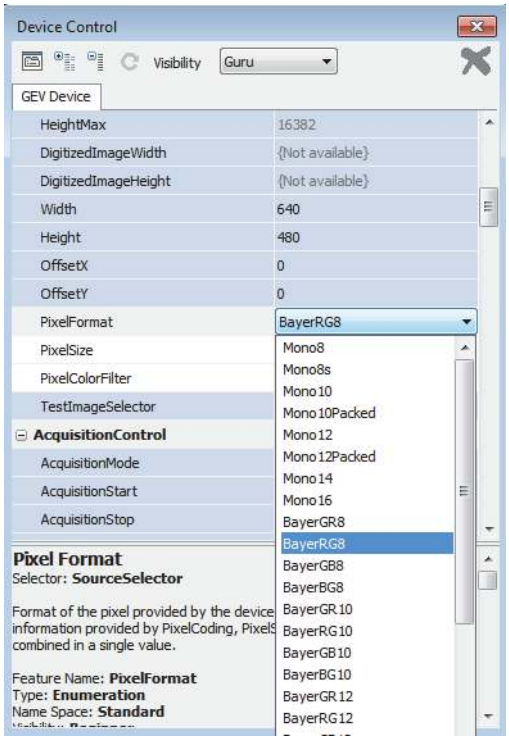
1. Start eBUS Player and connect to the embedded video interface.

   For more information, see "To start eBUS Player and connect to a device" on page 50.
2. Under **Parameters and Controls**, click **Device control**.
3. Under **SourceControl**, click the source that you want to configure.
4. Under **ImageFormatControl**, click a test pattern option in the **TestImageSelector** list.
5. Close the **Device Control** dialog box.

### To change the pixel format

1. Start eBUS Player and connect to the embedded video interface.

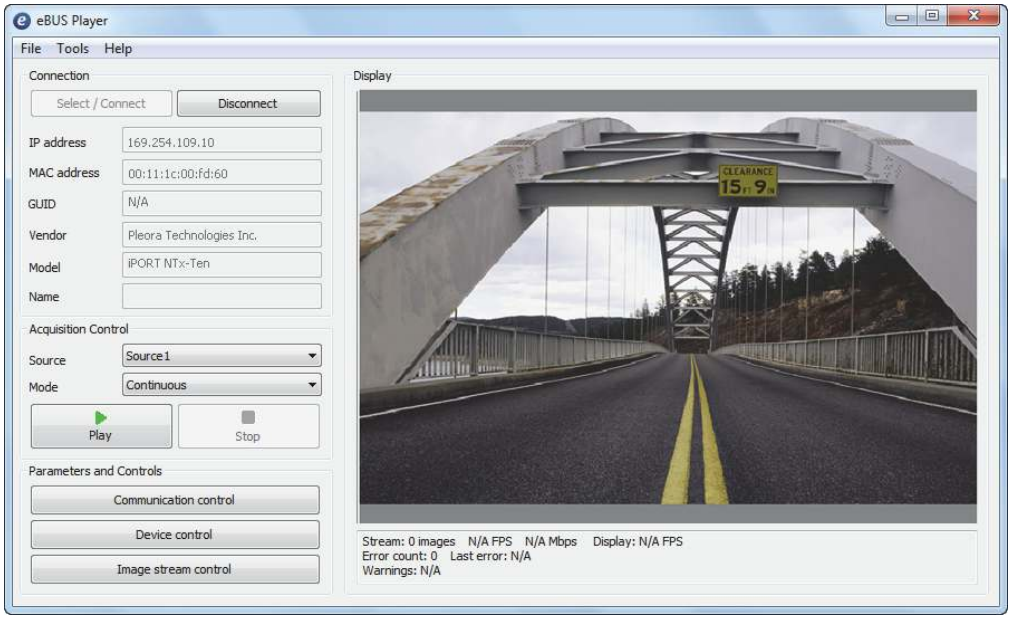
   For more information, see "To start eBUS Player and connect to a device" on page 50.
2. If images are streaming, click the **Stop** button
3. Under **Parameters and Controls**, click **Device control**.
4. Under **SourceControl**, click the source that you want to configure.
5. Under **ImageFormatControl**, set the **PixelFormat** feature to a color format, such as Bayer or RGB (by default the **PixelFormat** is set to **Mono8**).




Changes to the **PixelFormat** may affect the **DeviceScanType**, **SensorDigitizationTaps**, **DeviceTapGeometry**, **Width**, **Height**, **DigitizedImageWidth**, and **DigitizedImageHeight** features. When you change the **PixelFormat**, the embedded video interface may automatically adjust the other values to ensure the configuration is valid.

6. Close the **Device Control** dialog box.

7. Under **Acquisition Control**, click the source to which the camera you want to view is connected.
8. Click **Play** to see the image stream in color.




## To configure the image width and height

1. Start eBUS Player and connect to the embedded video interface.

   For more information, see "To start eBUS Player and connect to a device" on page 50.
2. If images are streaming, click the **Stop** button.
3. Under **Parameters and Controls**, click **Device Control**.
4. Under **SourceControl**, click the source that you want to configure.
5. Under **ImageFormatControl**, change the **Width** and **Height** to suit your camera.
   - If the embedded video interface does not perform tap reconstruction, configure the **Width** and **Height**.
   - If the embedded video interface does perform tap reconstruction. Configure the **Width**, **Height**, **DigitizedImageWidth** and **DigitizedImageHeight**.
6. Close the **Device Control** dialog box.

Changes to the **Width**, **Height**, **DigitizedImageWidth**, and **DigitizedImageHeight** may affect the **DeviceScanType**, **SensorDigitizationTaps**, **DeviceTapGeometry**, and **PixelFormat** features. When you change these features, the embedded video interface may automatically adjust the other values to ensure the configuration is valid.

# Configuring Camera Settings

The NTx-Ten Embedded Video Interface supports up to two cameras streaming image data. To ensure images are received properly, you must configure the general camera settings, which include specifying the sensor scan type (either areascan or linescan), selecting the number of taps for your camera, and selecting your camera's tap geometry. All of this information is provided by the camera manufacturer.

## Supported Camera Modes

The following table lists the supported camera modes and sub-modes.

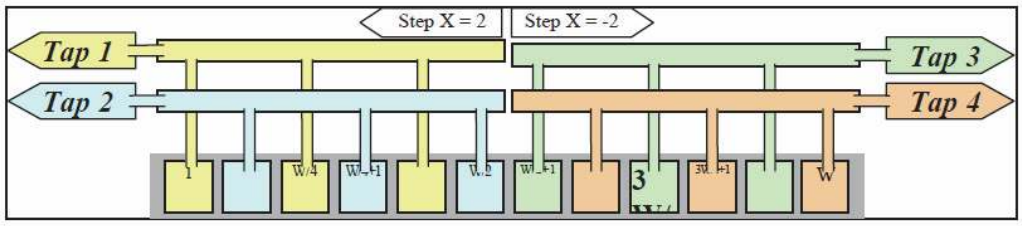
Table 18: Supported Camera Modes

| Mode | Sub-mode |
|---|---|
| Base | 8-bit x 1 tap<br>8-bit x 2 taps<br>10-bit x 1 tap<br>10-bit x 2 taps<br>12-bit x 1 tap<br>12-bit x 2 taps<br>14-bit x 1 tap<br>16-bit x 1 tap<br>24-bit RGB 8 |
| Medium | 8-bit x 4 taps<br>10-bit x 4 taps<br>12-bit x 4 taps<br>30-bit RGB 10<br>32-bit RGBA 8<br>36-bit RGB 12 |
| Single Source Dual Medium | 12-bit x 8 taps |

## Multiple Tap Support

For multi-tap cameras, pixels may not be received in order from the camera, depending on the camera's tap configuration. Using eBUS Player, you can specify the device tap geometry that corresponds to your camera, which allows the embedded video interface to reconstruct the pixel order. The following figure provides the supported tap geometry that requires reconstruction.

Figure 1: 2X2E (Linescan) Tap Geometry




When you select a tap geometry for which the embedded video interface performs tap reconstruction, you must provide the embedded video interface with information about the size of the image coming from the camera. This information allows the embedded video interface to send a portion of the image (an area of interest) to the receiver, when required.

## Supported Tap Geometries

The following table lists the supported tap geometries. Please note that the embedded video interface does not need to perform tap reconstruction for most of the supported tap geometries (that is, the taps are received in order from the camera and the embedded video interface does not need to reconstruct the order).

Table 19: Supported Tap Geometries

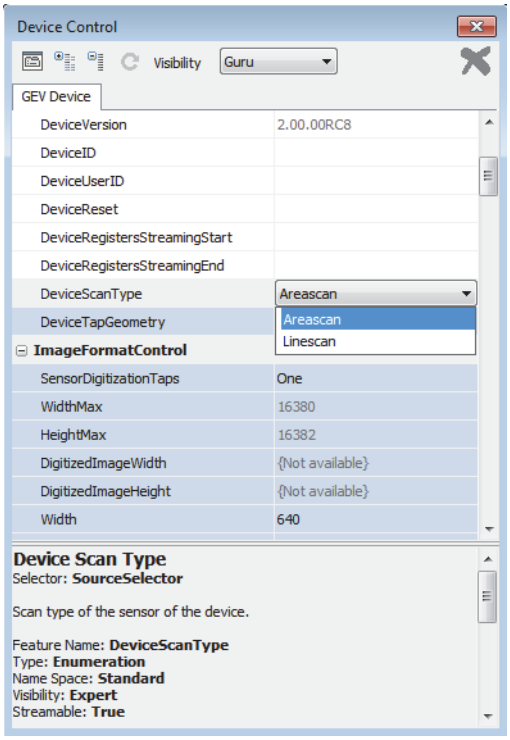
| Tap geometry | Number of taps | Tap reconstruction performed? | Scan type |
|---|---|---|---|
| Geometry_1X_1Y | 1 | Not required | Areascan |
| Geometry_1X2_1Y | 2 | Not required | Areascan |
| Geometry_1X | 1 | Not required | Linescan |
| Geometry_1X2 | 2 | Not required | Linescan |
| Geometry_1X4_1Y | 4 | Not required | Areascan |
| Geometry_1X4 | 4 | Not required | Linescan |
| Geometry_2X2E | 2 | Yes | Linescan |
| Geometry_1X8_1Y | 8 | Not required | Areascan |
| Geometry_1X8 | 8 | Not required | Linescan |

If your camera outputs a tap geometry that is not listed in the table above, you may need to perform tap reconstruction using your software application. For more information about tap geometry, refer to the *GenICam Standard Features Naming Convention* (Version 1.5.1 or later), available from the European Machine Vision Association at http://www.emva.org.

### To configure the embedded video interface scan type and tap geometry

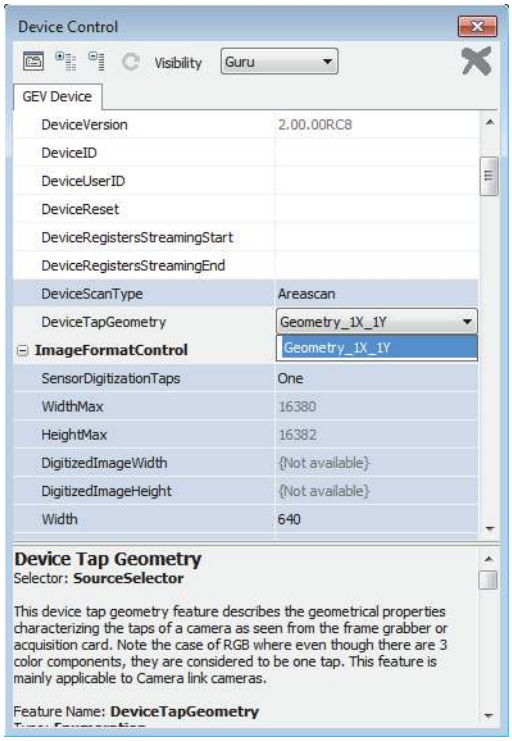
1. Start eBUS Player and connect to the embedded video interface.

   For more information, see "To start eBUS Player and connect to a device" on page 50.
2. If images are streaming, click the **Stop** button.
3. Under **Parameters and Controls**, click **Device control**.
4. Click **Expert** in the **Visibility** list.
5. Under **DeviceControl**, select a sensor scan type (areascan or linescan) in the **DeviceScanType** list.




6. Under **ImageFormatControl**, select the number of taps in the **SensorDigitizationTaps** list.

7. Under **DeviceControl**, select your camera's tap geometry in the **DeviceTapGeometry** list.




**DeviceScanType**, **SensorDigitizationTaps**, and **DeviceTapGeometry** are interrelated. When you change any of these values, the embedded video interface may automatically adjust the other values to ensure the configuration is valid.

These values are also affected by changes to the **Width**, **Height**, **DigitizedImageWidth**, **DigitizedImageHeight**, **SourceCount**, and **PixelFormat** features.

8. Close the **Device Control** dialog box.

# Implementing the eBUS SDK

You can create your own image acquisition software for the embedded video interface. Consult the *eBUS SDK C++ API Help file* and the *eBUS SDK .NET API Help* file for information about creating custom image acquisition software.

# Chapter 11

## Network Configurations

After you have connected to the embedded video interface and provided it with a unique IP address on your network, you can configure the embedded video interface for either unicast or multicast.

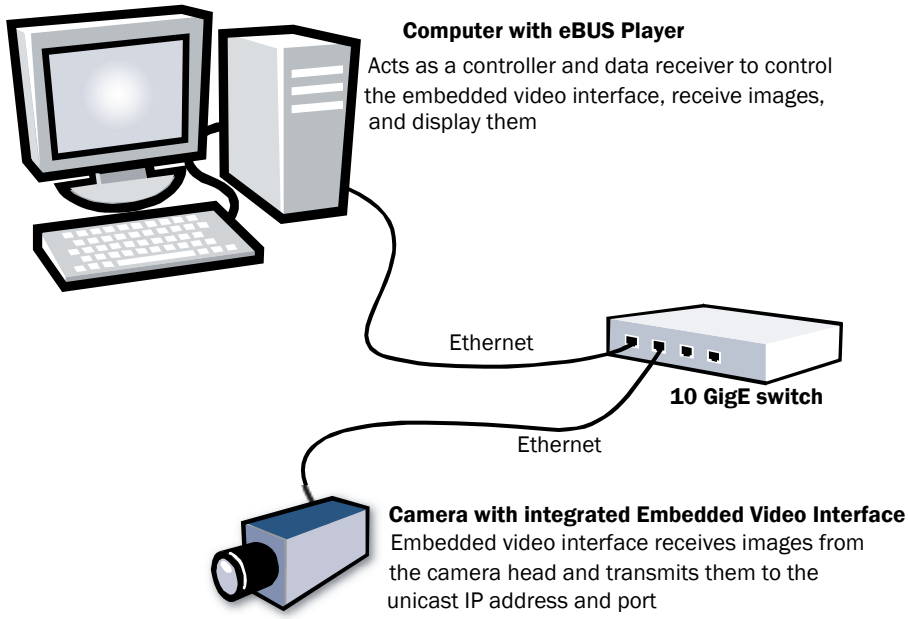
The following topics are covered in this chapter:

# Unicast Network Configuration

In a unicast configuration, an embedded video interface is connected to a 10 GigE switch that sends a stream of images over Ethernet to the computer. Alternatively, the embedded video interface can be connected directly to the computer.

The computer is configured as both a data receiver and controller, and serves as a management entity for the embedded video interface.

Figure 2: Unicast Network Configuration



## Required Items — Unicast Network Configuration

You require the following components to set up a unicast network configuration:

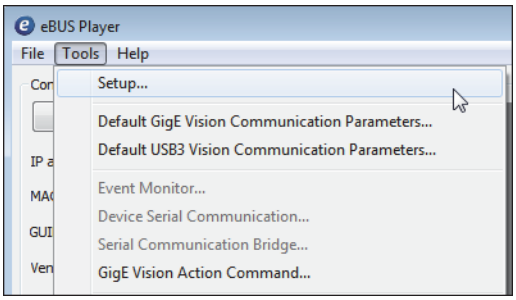
- iPORT NTx-Ten Embedded Video Interface
- 8-13 V power supply
- Fiber cable with SFP connectors (quantity: 1)
- Desktop computer with eBUS SDK installed
- Camera and cables

## Embedded Video Interface Configuration — Unicast Network Configuration

After you have connected and applied power to the hardware components, use eBUS Player to configure the embedded video interface.

### To configure the embedded video interface for a unicast network configuration

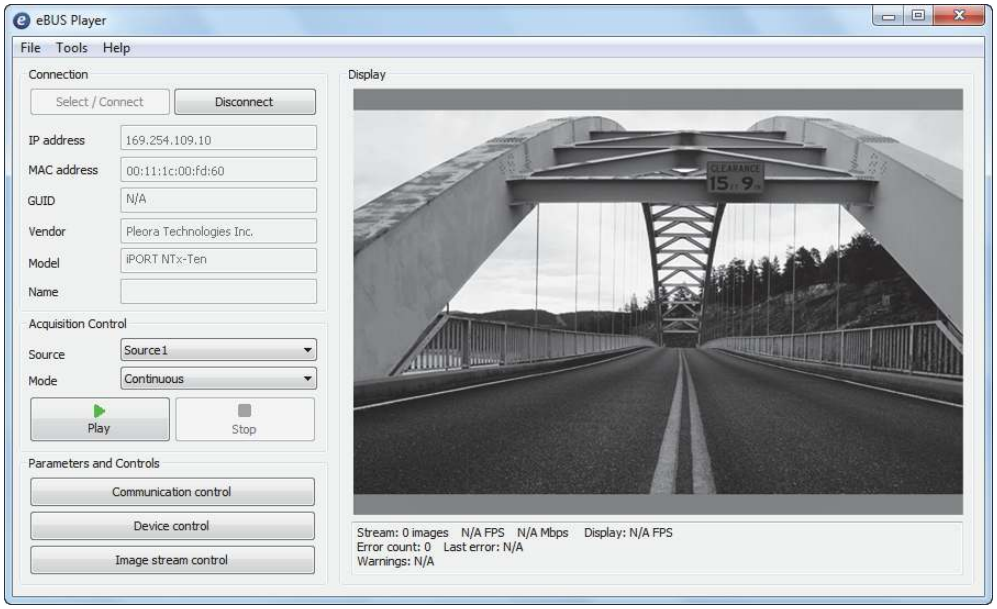
1. Start **eBUS Player**.
2. Click **Tools > Setup**.




3. Under **eBUS Player Role**, click **Controller and data receiver**.
4. Under **GigE Vision Stream Destination**, click **Unicast, automatic**.
5. Click **OK**.
6. Connect to the embedded video interface.

   For more information, see "To start eBUS Player and connect to a device" on page 50.

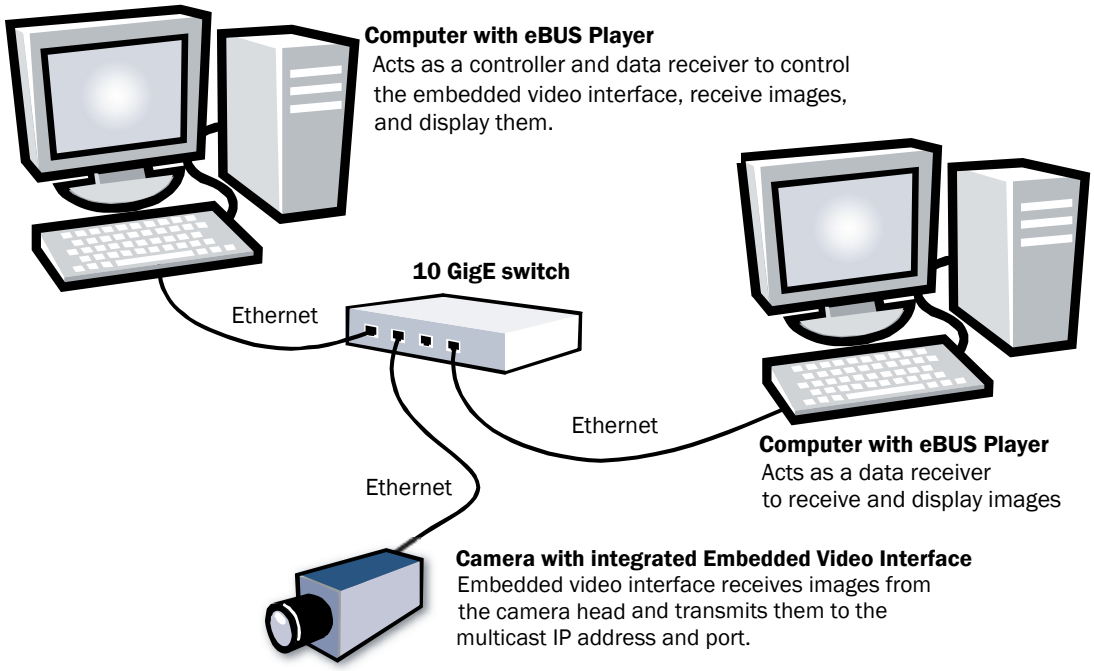
7. Click **Play** to view a live image stream.



# Multicast Network Configuration

In a multicast network configuration, the iPORT NTx-Ten Embedded Video Interface is connected to a 10 GigE switch, and sends a stream of images over Ethernet simultaneously to two or more computers running eBUS Player (or an application created with the eBUS SDK).

Figure 3: Multicast Network Configuration



## Required Items — Multicast Network Configuration

You require the following components to set up a multicast network configuration:

- iPORT NTx-Ten Embedded Video Interface
- 9-13 V power supply
- Fiber cable with SFP connectors (quantity: 3)
- 10 GigE switch (IGMP v2-compatible)
- Desktop computer (quantity: 2) with eBUS SDK installed
- Camera and cables

## Connecting the Hardware and Power

The following procedure explains how to connect the power, network, and data cables to the computers running eBUS Player and NTx-Ten Embedded Video Interface.

### To connect the network cables and apply power

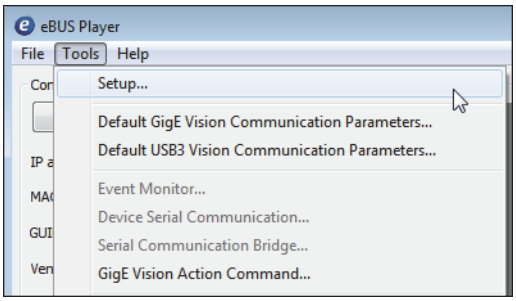
1. On the computer that is acting as a controller and data receiver with eBUS Player, connect a 10 GigE port on your NIC to a 10 GigE port on your 10 GigE switch. Repeat this step for the computer that is acting as a video receiver with eBUS Player.
2. Insert a fiber SFP+ module into the SFP+ Ethernet connector on the embedded video interface and connect the fiber cable to the module. Attach the other end to an available 10 GigE port on the 10 GigE switch.
3. Apply power to the devices.

## Configuring the iPORT NTx-Ten Embedded Video Interface for a Multicast Network Configuration

After you have connected and applied power to the hardware components, use eBUS Player to configure the iPORT NTx-Ten Embedded Video Interface for multicast configuration. Begin by configuring one eBUS Player application to act as a data receiver. Then, configure the embedded video interface to transmit images to a multicast IP address and port.

### To configure eBUS Player as a data receiver

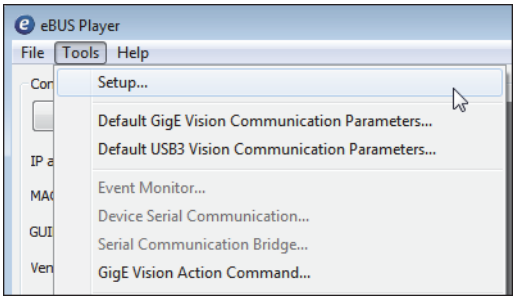
1. On the computer that is acting as a data receiver with eBUS Player, start **eBUS Player**.
2. Click **Tools > Setup**.




3. Under **eBUS Player Role**, click **Data receiver**.
4. Under **GigE Vision Stream Destination**, click **Multicast** and then specify a multicast address (for example, 239.192.1.1) and a streaming channel port (for example, 1042).
5. Click **OK**.
6. Now, configure the iPORT NTx-Ten Embedded Video Interface, as outlined in "To configure the iPORT NTx-Ten Embedded Video Interface for a multicast network configuration" on page 68.

### To configure the iPORT NTx-Ten Embedded Video Interface for a multicast network configuration

1. On the computer that is acting as a controller and data receiver with eBUS Player, start eBUS Player.
2. Click **Tools** > **Setup**.

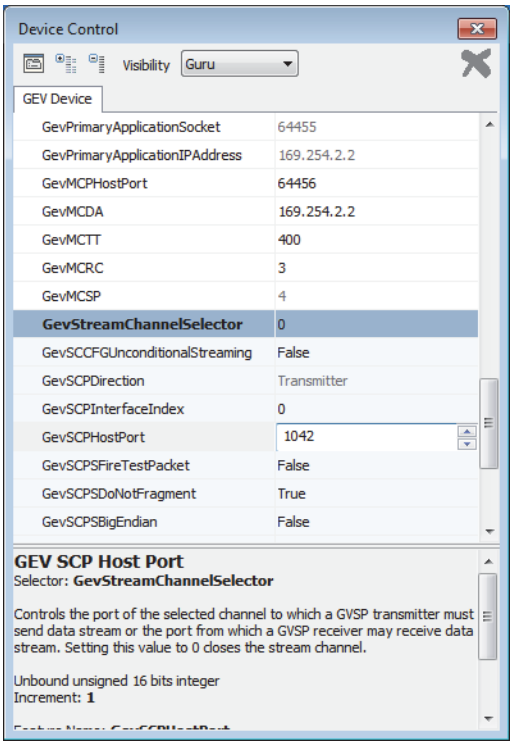



3. Under **eBUS Player Role**, click **Controller and data receiver**.
4. Under **GigE Vision Stream Destination**, click **Multicast** and enter the **IP address** and **Port** number.

   The address and port must be identical to that configured for the receiver in step 4 of "To configure eBUS Player as a data receiver" on page 67.
5. Click **OK**.
6. Connect to the iPORT NTx-Ten Embedded Video Interface.

   For more information, see "To start eBUS Player and connect to a device" on page 50.
7. Under **Parameters and Controls**, click **Device control**.
8. Click **Guru** in the **Visibility** list.

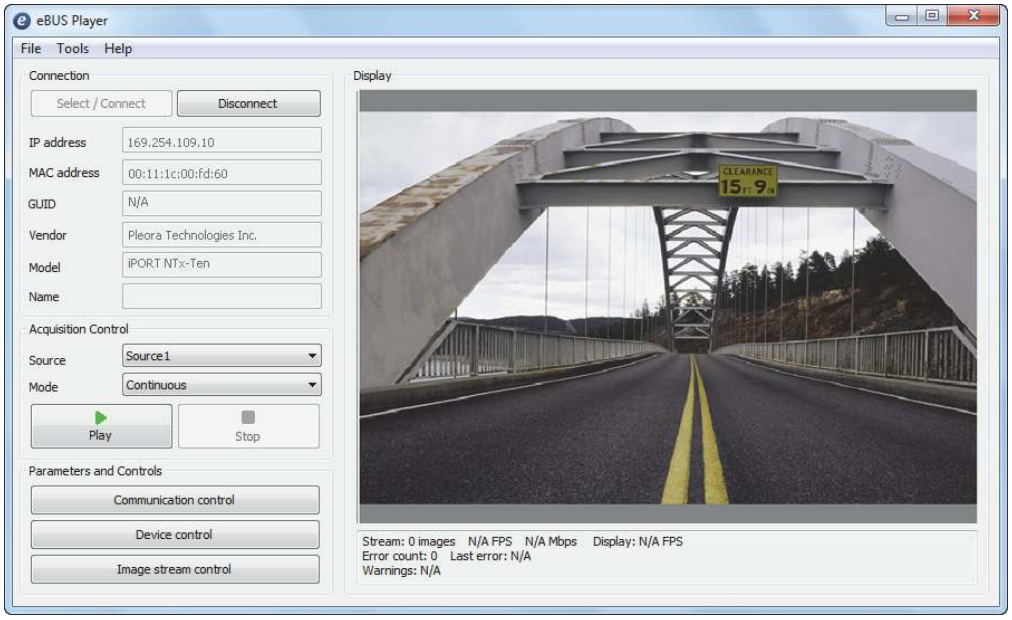
9. Under **TransportLayerControl**, ensure that the port in the **GevSCPHostPort** field and the multicast IP address in the **GevSCDA** field are correct. They are configured automatically to the values set in step 4 of this procedure.



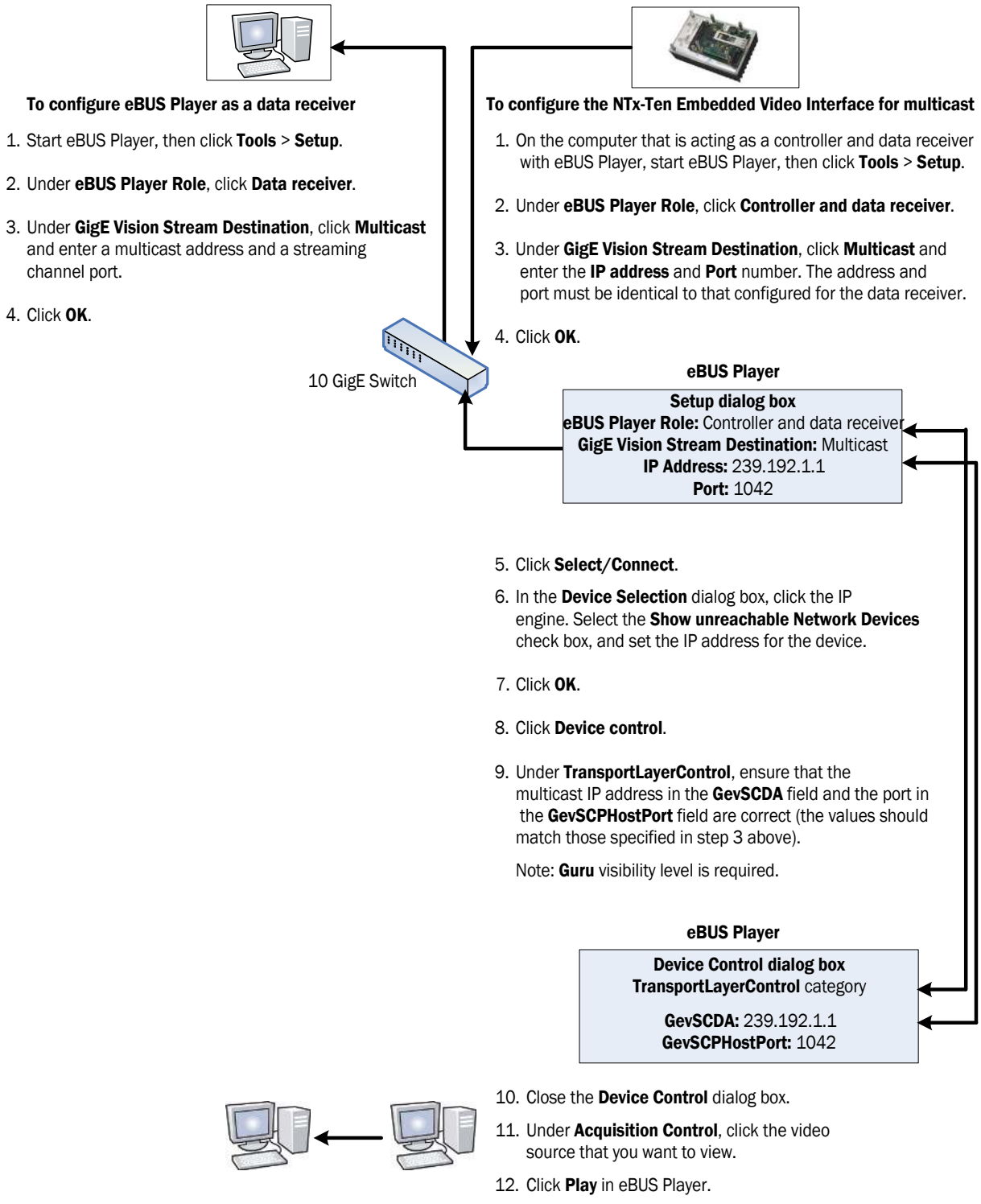

10. When you follow the instructions in this section, eBUS Player automatically configures the embedded video interface to send images from **Source1** to the default multicast group. To send images from a particular source to a specific multicast group, select the stream you want to configure in the **GevStreamChannelSelector**. Then, specify the multicast group information in the **GevSCPHostPort** and **GevSCDA** fields. In the **GevStreamChannelSelector**, **0** corresponds to **Source1** and **1** corresponds to **Source2**.
11. Close the **Device Control** dialog box.
12. Under **Acquisition Control**, click the source to which the camera you want to view is connected.

**13.** Click **Play** to view the source image stream on the computer.

The multicast image is shown on the computer and the display monitor receiver, as shown below.




An overview of the steps for the simple multicasting of the iPORT NTx-Ten Embedded Video Interface is shown in the following figure.



# Chapter 12

## System Troubleshooting

This chapter provides you with troubleshooting tips and recommended solutions for issues that can occur during configuration, setup, and operation of the embedded video interface.

Not all scenarios and solutions are listed here. You can refer to the Pleora Technologies Support Center at supportcenter.pleora.com for additional support and assistance.

The Pleora Technologies Support Center can help you to learn more about integrating Pleora Technologies products. Use keywords to search the Pleora Technologies knowledge database for solutions and suggestions to optimize and troubleshoot Pleora Technologies products. The knowledge database includes a description of the issue and the suggested solution for your search results.

Details for creating a customer account are available at the Pleora Technologies Support Center.

Refer to the product release notes that are available at the Pleora Technologies Support Center for known issues and other product features.

# Troubleshooting Tips

The scenarios and known issues listed in the following table are those that you might encounter during the setup and operation of your embedded video interface. Not all possible scenarios and errors are presented. The symptoms, possible causes, and resolutions depend upon your particular network, setup, and operation.

If you perform the resolution for your issue and the issue is not corrected, we recommend you review the other resolutions listed in this table. Some symptoms may be interrelated.

Table 20: Troubleshooting Tips

| Symptom | Possible cause | Resolution |
|---|---|---|
| SDK cannot detect or connect to the embedded video interface | Power not supplied to the embedded video interface | Both the detection and connection to the embedded video interface will fail if power is not supplied to the device.<br>Verify that the Link LED is active. For information about the LEDs, see "Status LEDs" on page 23.<br>Re-try the connection to the embedded video interface with eBUS Player. |
| | Embedded video interface not connected to the network | Verify that the Link LED is active. If this LED is illuminated, check the LEDs on your network switch to ensure the switch is functioning properly. If the problem continues, connect the embedded video interface directly to the computer to verify its operation. For information about the LEDs, see "Status LEDs" on page 23. |
| | Embedded video interface and computer are not on the same subnet | Images might not appear in eBUS Player if the embedded video interface and the computer running eBUS Player are not on the same subnet. Ensure that these devices are on the same subnet. In addition, ensure that these devices are connected using valid gateway and subnet mask information. You can view the embedded video interface IP address information in the **Available GigE Vision Devices** list in eBUS Player. A red icon appears beside the device if there is an invalid IP configuration. |
| SDK is able to connect, but no images appear in eBUS Player.<br>In a multicast configuration, images appear on a display monitor connected to a vDisplay embedded video interface but do not appear in eBUS Player. | In a multicast configuration, the embedded video interface may not be configured correctly | Images might not appear on the display if you have not configured the embedded video interface for a multicast network configuration. The embedded video interface and all multicast receivers must have identical values for both the **GevSCDA** and **GevSCPHostPort** attributes in the **TransportLayerControl** section. For more information, see "Multicast Network Configuration" on page 65. |
| | In a multicast configuration, your computer's firewall may be blocking eBUS Player | Ensure that eBUS Player is allowed to communicate through the firewall. |
| | Anti-virus software or firewalls blocking transmission | Images might not appear in eBUS Player because of anti-virus software or firewalls on your network. Disable all virus scanning software and firewalls, and re-attempt a connection to the embedded video interface with eBUS Player. |

Table 20: Troubleshooting Tips (Continued)

| Symptom | Possible cause | Resolution |
|---|---|---|
| Dropped packets: eBUS Player, NetCommand, or applications created using the eBUS SDK | Insufficient computer performance | The computer being used to receive images from the embedded video interface may not perform well enough to handle the data rate of the image stream. The eBUS Universal Pro driver reduces the amount of computer resources required to receive images, and is recommended for applications that require high throughput. Should the application continue to drop packets even after the installation of the eBUS Universal Pro driver, a computer with better performance may be required. |
| | Insufficient NIC performance | The NIC being used to receive images from the embedded video interface may not perform well enough to handle the data rate of the image stream. For example, the bus connecting the NIC to the CPU may not be fast enough, or certain default settings on the NIC may not be appropriate for reception of a high-throughput image stream. Examples of NIC settings that may need to be reconfigured include the number of Rx Descriptors and the maximum size of Ethernet packets (jumbo packets). Additionally, some NICs are known to not work well in high-throughput applications. |
| Black bars appear on the sides of the images | Camera does not output images using the full image size | In eBUS Player, adjust the **Width**, **Height**, **DigitizedImageWidth**, **DigitizedImageHeight**, and image offset features until the black bars no longer appear. |
| Images are not properly sized or are not properly positioned in the window | Image width, height, or offset not set correctly, based on the **InputVideoFormat** feature | In eBUS Player, adjust the **Width**, **Height**, **DigitizedImageWidth**, **DigitizedImageHeight**, **OffsetX**, and **OffsetY** to the correct value, based on the **InputVideoFormat** settings that you configured. |
| Exclamation marks  appear beside the **Width**, **Height**, **DigitizedImageWidth**, **DigitizedImageHeight**, **OffsetX**, or **OffsetY** features in eBUS Player | | |

# Changing to the Backup Firmware Load

In some cases, you may need to change from the main firmware load to the backup firmware load. To do this, you can use the slide switch, shown in the following image.

To access the slide switch, peel back the small, protective plastic sheet that covers the switch.

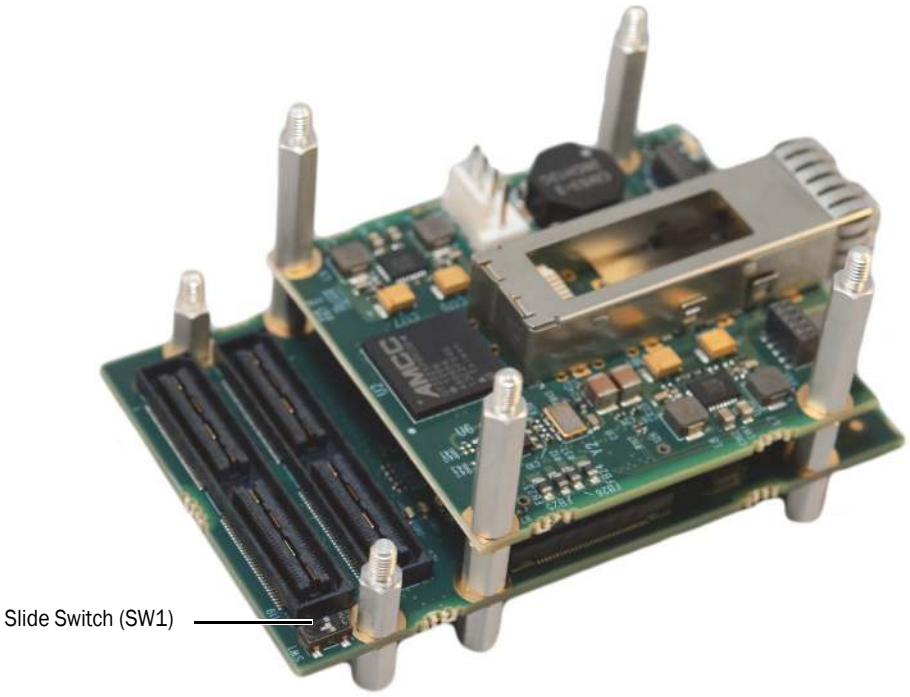



Table 21: Slide Switch Settings

| SW1 slide switch | FPGA load |
|---|---|
| Off | Main load |
| On | Backup load |

## Using the FPGA Load Selection Pin

The FPGA load selection pin (pin 59, located on connector J5) can be driven externally to force the backup firmware load to run. To do this, the slide switch (SW1) must be left in the Off position.

With the SW1 switch in the Off position and the FPGA load selection pin left open, the main firmware load will run; if the pin is grounded, the backup load will run.

If you only want to use the slide switch SW1 to control which firmware load runs, the FPGA load selection pin must be left open.

The status of the SW1 slide switch, or the FPGA load selection pin, is sampled once at power-up and as a result, the state of the switch or pin must be set prior to device power-up.

# Chapter 13

## Reference: Mechanical Drawings and Material List

This chapter provides the mechanical drawings, and also provides a list of connectors and cables, with corresponding manufacturer details.

The following topics are covered in this chapter

# Mechanical Drawings

The mechanical drawings in this section provide the embedded video interface's dimensions, features, and attributes. All dimensions are in millimeters.

3D models in STEP format are available on the Pleora Support Center at supportcenter.pleora.com. For more information, see "Technical Support" on page 85.

Figure 4: OEM Board Front View

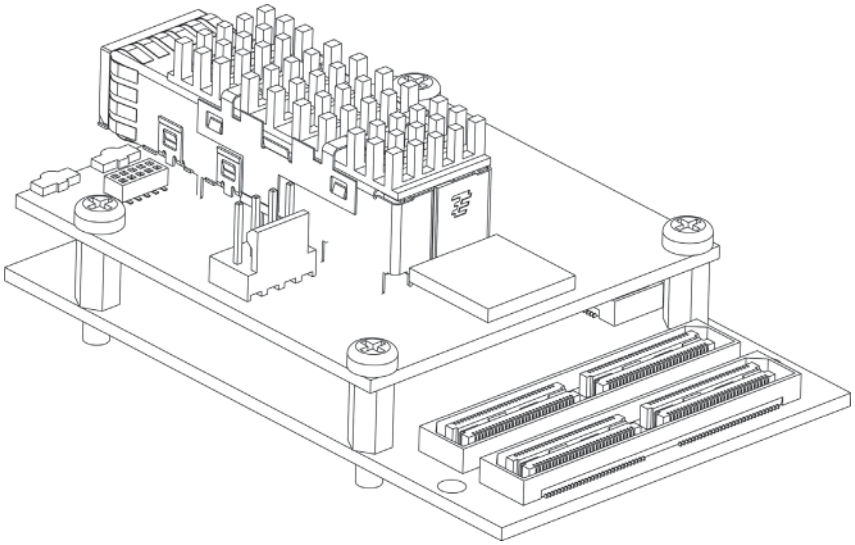


Figure 5: OEM Board Set Top View with Dimensions

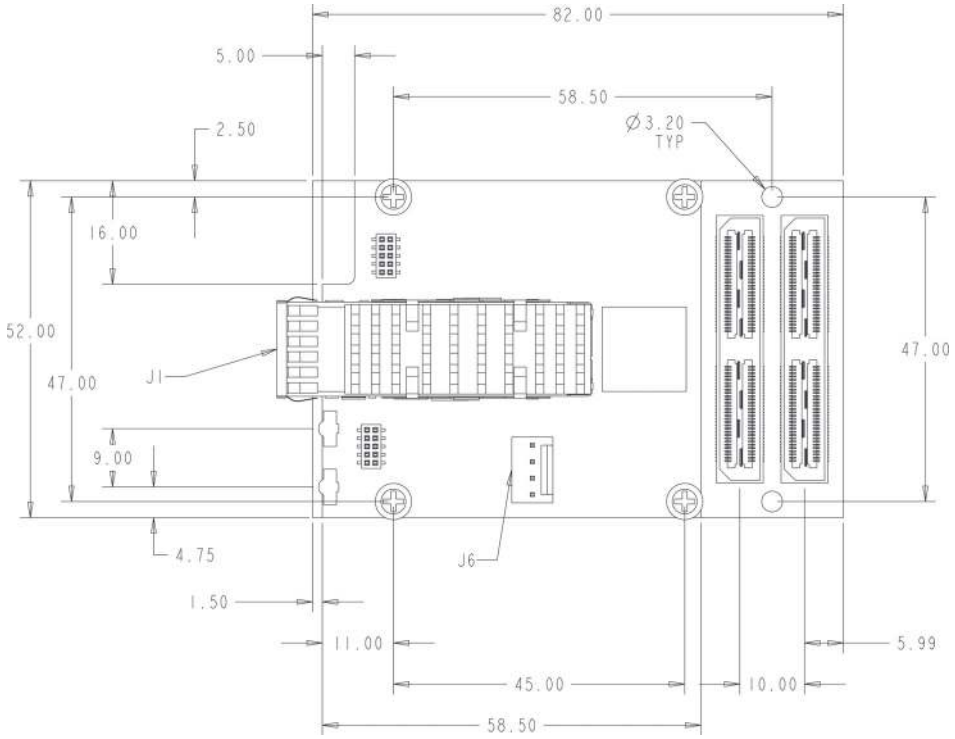



Figure 6: OEM Board Set Front View with Dimensions

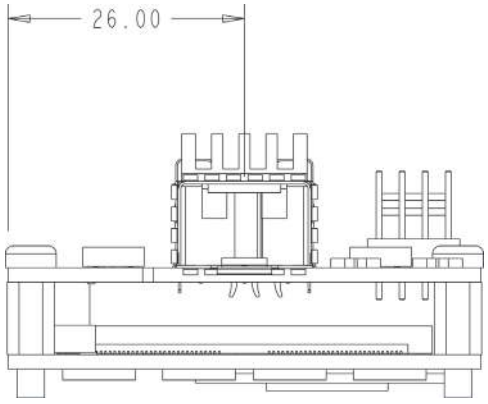


Figure 7: OEM Board Set Side View with Dimensions

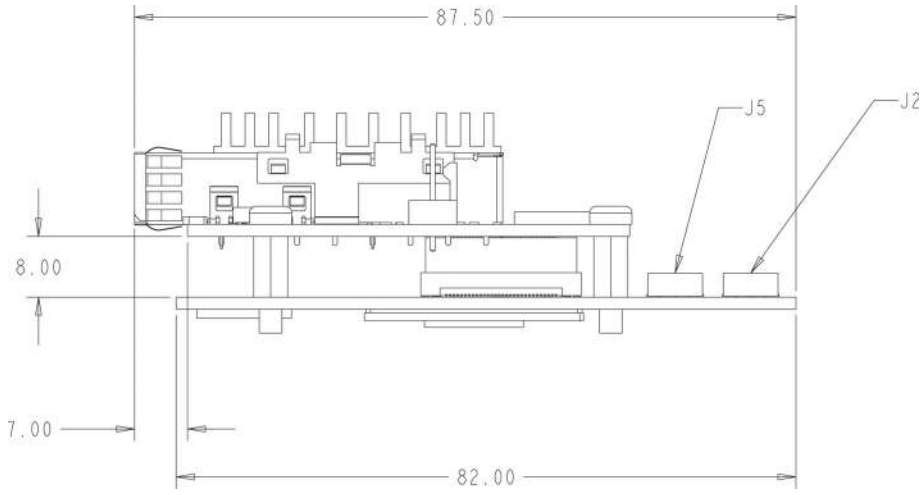
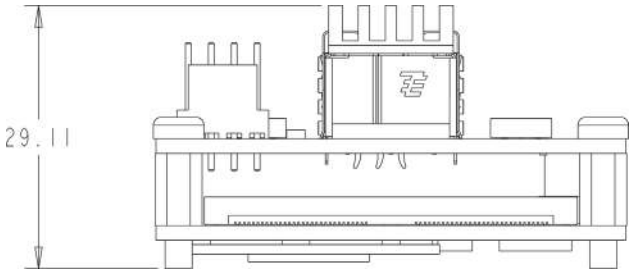



Figure 8: OEM Board Set Back View with Dimensions



# Material List

The connector summary for the embedded video interface is provided in the following table.

Table 22: NTx-Ten Embedded Video Interface Connector Summary

| ID | Description | Manufacturer Part Number | Manufacturer |
|---|---|---|---|
| J1 | SFP+ Ethernet connector | 1888247-2 | Tyco Electronics |
| M1 | SFP+ Cage | 2007194-2 | Tyco Electronics |
| J6 | 4-pin power connector | 22-23-2041 | Molex |
| J2, J5 | 120-pin user circuitry connector, female | QSH-060-01-F-D-A | Samtec |

Table 23: Mate to Connector Summary

| Mate connector | Description | Manufacturer Part Number | Manufacturer |
|---|---|---|---|
| Mate to J1 | Any SFP+ connector | | |
| Mate to J6 | Crimp housing (x1) | 22-01-3047 | Molex |
| | Crimp terminals (x4) | 8550102 | Molex |
| Mate to J2/J5 | 120-pin connector, male (8mm board-to-board spacing) | QTH-060-02-F-D-A | Samtec |

# Chapter 14

## Technical Support

On the Pleora Support Center, you can:

- Download the latest software and firmware.
- Log a support issue.
- View documentation for current and past releases.
- Browse for solutions to problems other customers have encountered.
- Read knowledge base articles for information about common tasks.

### To visit the Pleora Support Center

- Go to supportcenter.pleora.com and click **Support Center**.
  If you have not registered yet, you are prompted to register.
  Accounts are usually validated within one business day.