eBUS SDK for Linux Quick Start Guide

Pleora Technologies

Table of Contents

Document Number and Revisions

Document ID: QSG-0011

Revision	Reason for Revision	Date
17.0	Final Release with eBUS SDK 6.5.0	May 2024
16.0	Final Release with eBUS SDK 6.4.0	February 2024

1 About this Guide

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

1.1 About the eBUS SDK for Linux

eBUS SDK is built on a single API to receive video over GigE, 10 GigE, and USB 3.0 that is portable across Windows, Linux, and macOS operating systems. With an eBUS SDK Seat License, designers can develop production-ready software applications in the same environment as their end-users, and quickly and easily modify applications for different media, while avoiding supporting multiple APIs from various vendors. Compared to camera vendor provided SDKs, eBUS frees developers from being tied to a specific camera, and instead they can choose the device that is best for the application.

eBUS Edge for Sensor Devices

eBUS Edge is a software implementation of a full device level GigE Vision transmitter, without requiring any additional hardware. Adding eBUS Edge to a CPU's software stack turns it into a fully compliant GigE Vision device that supports image transmission and enables the device to respond to control requests from a host controller. eBUS Edge is GigE Vision and GenICam compliant, meaning end-users can use any standards- compliant third-party image processing system. eBUS Edge currently supports the GigE Vision standard.

eBUS Receive for Host Applications

eBUS Receive manages high-speed reception of images or data into buffers for hand-off to the end application for further analysis. Developers can write applications that run on a host computer to seamlessly control and configure an unlimited number of GigE Vision or USB3 Vision and GenICam compliant sensors.

The eBUS Universal Pro driver reduces CPU usage when receiving images or data, leaving more processing power for analysis and inspection applications while helping meet latency and throughput requirements for real-time applications. The eBUS Universal Pro driver is easily integrated into third-party processing software to bring performance advantages to end-user applications.

1.2 What this Guide Provides

This guide provides you with the steps to use the eBUS SDK on the Linux operating system, on a Linux x86_64, or ARM platform. This guide is intended for novice Linux users, although advanced Linux users may be interested in some of the eBUS SDK-specific elements of this guide.

1.3 Documented Product Version

This guide covers Release 6.5 of the eBUS SDK. The features and functionality documented in this guide may vary if you are using an earlier or later version of the eBUS SDK.

1.4 Related Documents

eBUS SDK Related Documents

Guides are complemented by the following Pleora Technologies documents, which are available on the Pleora Technologies Support Center (supportcenter.pleora.com):

- eBUS SDK C++ API Quick Start Guide
  This guide provides you with the information you need to install the eBUS SDK (which lets you use the eBUS SDK C++ API) and an overview of the system requirements.

- Docker Support for eBUS SDK User Guide
  This guide provides instruction on how to get started with using eBUS SDK in a Docker environment. It also provides basic instruction on how to set up Docker on a Ubuntu-based system.

- Getting Started with eBUS Edge
  The eBUS Edge API (introduced in eBUS SDK 6.0) allows developers to create a software-based GigE Vision transmitter device. eBUS Edge is fully compliant with GigE Vision and GenICam and will work with any GigE Vision and GenICam compliant third-party image processing systems or software.

- eBUS Edge on OpenSTLinux (ST Micro MP2 Platform) Quick Start Guide
  This guide provides you with the steps to use the eBUS Edge on the ST Micro MP2 Platform running OpenSTLinux. This guide is intended for novice Linux users, although advanced Linux users may be interested in some of the eBUS SDK-specific elements of this guide for cross-compiling eBUS Edge application using eBUS SDK.

- eBUS SDK Licensing
  This knowledge base article explains the eBUS SDK license structure, explains how to obtain a license, and provides activation instructions. If you are experiencing difficulty activating your license, please review the troubleshooting steps at the end of this publication.

- eBUS SDK for Linux Quick Start Guide
  This guide provides you with the steps to use the eBUS SDK on the Linux operating system, on a Linux x86_64, or ARM platform. This guide is intended for novice Linux users, although advanced Linux users may be interested in some of the eBUS SDK-specific elements of this guide.

- eBUS SDK .NET API Quick Start Guide
  This guide provides you with instructions for compiling and using the sample code are provided, along with an overview of the basic calls that can be used to build custom applications using the eBUS SDK .NET API.

- eBUS Player User Guide
  This guide provides in-depth details about setting up and using the eBUS Player software application to control your GigE Vision or USB3 Vision compliant video transmitters (cameras) and receivers.

- eBUS Player Quick Start Guide
  This quick start guide provides you with the information you need to start using the eBUS Player application, which lets you control the parameters of your GigE Vision or USB3 Vision compliant device and lets you view imaging video and data.

- eBUS SDK Python API Quick Start Guide
  This guide provides you with the information you need to install the eBUS SDK (which lets you use the eBUS SDK Python API) and an overview of the system requirements.

- eBUS SDK on Raspberry PI 4/5 Quick Start Guide
  This guide provides you with the steps to use the eBUS SDK on a Raspberry PI 4/5. This guide is intended for novice Raspberry PI 4/5 users, although advanced Raspberry PI 4/5 users may be interested in some of the eBUS SDK-specific elements of this guide.

- Supported Software Ecosystem
  This support summary provides an overview of the supported protocols, operating systems, ARM platforms, development environments, and drivers. Pleora is currently supporting eBUS SDK 5.0 and later.

2 Installing the eBUS SDK for Linux

This chapter provides system requirements and steps that you will use to install the eBUS SDK for Linux.

2.1 System Requirements

Ensure the computer on which you install the eBUS SDK meets the following recommended requirements:

> You can access installation files from the Pleora Support Center at supportcenter.pleora.com.

At least one Gigabit Ethernet NIC (if you are using GigE Vision devices) or at least one USB 3.0 port (if you are using USB3 Vision devices).

An appropriate compiler or integrated development environment (IDE):

- Visual Studio 2019, 2017, 2015, 2013, 2012, and 2010.
- Sample project files (.vcxproj) are compatible with Visual Studio 2012 (and later). When you open them with Visual Studio for the first time, you have the option of upgrading

> For supported USB 3.0 host controller chipsets, consult the eBUS SDK Release Notes, available on the Pleora Support Center.

> Depending on the incoming and outgoing bandwidth requirements, as well as the performance of each NIC, you may require multiple NICs. For example, even though Gigabit Ethernet is full duplex (that is, it manage 1 Gbps incoming and 1 Gbps outgoing), the computer's PCIe bus may not have enough bandwidth to support this. This means that while your NIC can — in theory — accept four cameras at 200 Mbps each incoming, and output a 750 Mbps stream on a single NIC, the NIC you choose may not support this level of performance.

One of the following operating systems:

- For Microsoft Windows
  - Microsoft Windows 11, 64-bit
  - Microsoft Windows 10, 32-bit or 64-bit
  - Microsoft Windows 8.1, 32-bit or 64-bit
  - Microsoft Windows 7 with Service Pack 1 (or later), 32-bit or 64-bit
- For the x86 Linux platform:

    - Ubuntu 22.04 LTS 64-bit: Qt 5.15.3
    - Ubuntu 20.04 LTS 64-bit: Qt 5.12.8
    - Ubuntu 18.04 LTS 64-bit Qt 5.9.5
    - RHEL 8.7, 64-bit: Qt 5.15.3
    - CentOS Stream 8, 64-bit: Qt 5.15.3
- For the Linux for ARM platform:
    - NVIDIA Jetson TX2, Jetson Nano, Jetson Xavier NX, Jetson AGX Xavier, Jetson TX2 NX running JetPack 4.6 (Ubuntu 18.04)
    - NVIDIA Jetson AGX Xavier, Jetson Xavier NX, Jetson AGX Orin, Jetson Orin NX running Jetpack 5.1.x which x is between 0 and 3. (Ubuntu 20.04)
    - Raspberry Pi 4 , Raspberry Pi 5 running Raspberry Pi OS (64 bits) (Debian aarch64 GNU/Linux 12 bookworm)

According to the Linux/ARM platform, we provide eBUS Python for the stock Python version of the OS:

- Ubuntu 22.04 Desktop (64-bit), eBUS Python for Python 3.10 is installed with the SDK
- Ubuntu 20.04 Desktop (64-bit), eBUS Python for Python 3.8 is installed with the SDK
- Ubuntu 20.04 for ARM (64-bit), eBUS Python for Python 3.8 is installed with the SDK
- Ubuntu 18.04 Desktop (64-bit), eBUS Python for Python 3.6 is installed with the SDK
- Ubuntu 18.04 for ARM (64-bit), eBUS Python for Python 3.6 is installed with the SDK
- RHEL 8 (64-bit), eBUS Python for Python 3.6 is installed with the SDK
- CentOS 8 Stream (64-bit), eBUS Python for Python 3.6 is installed with the SDK
- Raspberry Pi Desktop (64-bit), eBUS Python for Python 3.11 is installed with the SDK.

> For non-default Python versions for different Linux ARM/x86 platforms, consult your Pleora support representative.

Ensure that you uninstall any previous versions of eBUS SDK from your machine before installing eBUS SDK 6.x.

2.2 Installing the eBUS SDK for Linux

Use the installation package to install the eBUS SDK and eBUS Runtime for Linux.

Install required dependencies:

1. The following prerequisites are critical to be able to save MP4 video along with other functionality in eBUS. They are tightly bound to eBUS SDK so they need to be installed regardless of whether or not MP4 video is being saved.
   Note that there are additional steps required to enable MP4 video saving in eBUS Player (these are described later on in this section).
   - For Ubuntu 18.04 (64-bit) execute the command:
     - sudo apt-get install libavcodec57
   - For Ubuntu 20.04 (64-bit) execute the command:
     - sudo apt-get install libavcodec58
   - For Ubuntu 22.04 (64-bit) execute the command:
     - sudo apt-get install libavcodec58
   - For RHEL8 execute the following commands:
     - sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
     - sudo yum install https://download1.rpmfusion.org/free/el/rpmfusion- free-release-8.noarch.rpm https://download1.rpmfusion.org/nonfree/el/rpmfusion-nonfree-release- noarch.rpm
     - sudo yum config-manager --set enabled powertools
     - sudo yum install ffmpeg
   - For CentOS 8 Stream:
     - sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
     - sudo yum install https://download1.rpmfusion.org/free/el/rpmfusion- free-release-8.noarch.rpm https://download1.rpmfusion.org/nonfree/el/rpmfusion-nonfree-release- noarch.rpm
     - sudo subscription-manager repos --enable codeready-builder-for-rhel- 8x86_64-rpms
     - sudo yum install ffmpeg
   - If not already installed, see "To install the eBUS Runtime packages"
2. Install Qt5 on your system
   If Qt is currently installed on your system, ensure that you have the correct version and append the QT_Install_Path/bin on the **$PATH**.
   - For Ubuntu 18.04 and 20.04 (64-bit) execute the command:
     - sudo apt-get install qt5-default
   - For Ubuntu 22.04 (64-bit) execute the command:
     - sudo apt-get install qtbase5-dev qt5-qmake
   - For On RHEL8/CentOS Stream 8 execute the command (should be run as a super user):
     - yum install qt5-qtbase-devel
3. For Ubuntu 22.04, 20.04, or 22.04, execute the command**:**

- sudo apt-get install build-essential python3-numpy

4. For Ubuntu 22.04, execute the following command to update the compiler to build the eBUS driver when the linux kernel version is 6.5 or later.
   - sudo apt-get install gcc-12

> You must install build-essential before installing eBUS SDK in order to allow the compilation of the eBUS Universal Pro driver during the eBUS SDK installation. Build-essential is also required to compile samples.

5. For CentOS 8 Stream and RedHat 8, as super user, execute the command:
   - yum install elftutil-libelf-devel
   - yum install python3-numpy

Install the eBUS SDK

1. Copy the eBUS SDK installation package to your workstation or embedded The installation package is available for download at supportcenter.pleora.com[1].
2. From the terminal, execute one of the following commands.
   The command varies depending on the distribution you are using.
   - Ubuntu x86_64:
     - sudo dpkg -i eBUS_SDK_<*distribution_targeted*>-<6.x.x>-<SDK build #>.deb
     - Where <*distribution_targeted*> can be:
       - Ubuntu-22.04 x86_64
       - Ubuntu-20.04 x86_64
       - Ubuntu-18.04 x86_64
   - For ARM platforms:
     - sudo dpkg -i eBUS_SDK_<distribution_targeted>-<6.x.x>-<SDK build #>.deb
     - Where <*distribution_targeted*> can be:
       - For eBUS SDK 4.0 through eBUS SDK 6.2:
         - linux-aarch-arm
       - For eBUS SDK 6.3 (and later):
         - linux-aarch64-arm
   - On RHEL/CentOS:
     - sudo rpm -i eBUS_SDK <*distribution targeted*>-x86_64-<6.x.x>-<SDK build #>.rpm
     - Where <*distribution_targeted*> can be:

1 http://supportcenter.pleora.com

- RHEL-CentOS (for RedHat 8)
- CentOS-RHEL (for CentOS 8 Stream)
  If any components failed to install, see the notes at the end of this procedure.

> ✅ For RHEL8 or CentOS 8 Stream, the firewall needs to be disabled in order to connect to a GigE Vision device over the network.
> As super user, execute the command:
> **systemctl disable firewalld**

3. We recommend that you reboot your workstation or embedded computer to ensure that the correct environment variables are set at The eBUS SDK is installed in the following directory:
   opt/pleora/ebus_sdk/<*distribution_targeted*>/
   For example: /opt/pleora/ebus_sdk/Ubuntu-20.04-x86_64

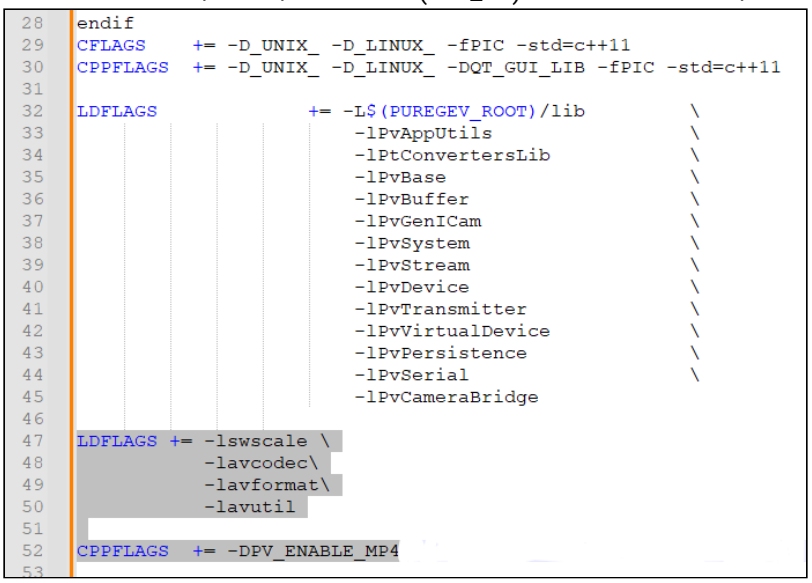
> 💡 If there is a failure to compile and install the eBUS Universal Pro driver due to permissions or missing dependencies, you can manually compile and install the driver. For more information, see “To manually install the eBUS Universal Pro for Ethernet Filter Driver” .

> ✅ If the eBUS SDK installation is not successful, your system may be missing the required Linux kernel headers.
> For more information, see “Error message: Cannot find the files to build kernel module in this PC".

> ✅ The following step is only needed if the eBUS SDK application requires MP4 video saving functionality.

4. On RHEL8/CentOS 8 Stream:
   - sudo yum install ffmpeg-devel
5. Enabling MP4 Video Saving in eBUS
   - Navigate to /opt/pleora/ebus_sdk/<*distribution_targeted*>/samples
   - Make a copy of the following directory, as a backup for the original source files:
     /opt/pleora/ebus_sdk/<distribution_targeted>/share/samples
   - Navigate to the directory that contains the copy of the sample code that you have "write"
   - Edit Makefile by adding the following highlighted code in the locations shown below:

- For Ubuntu 18.04/20.04/22.04 LTS (x86_64) and JetPack 4.6.3/5.1.x



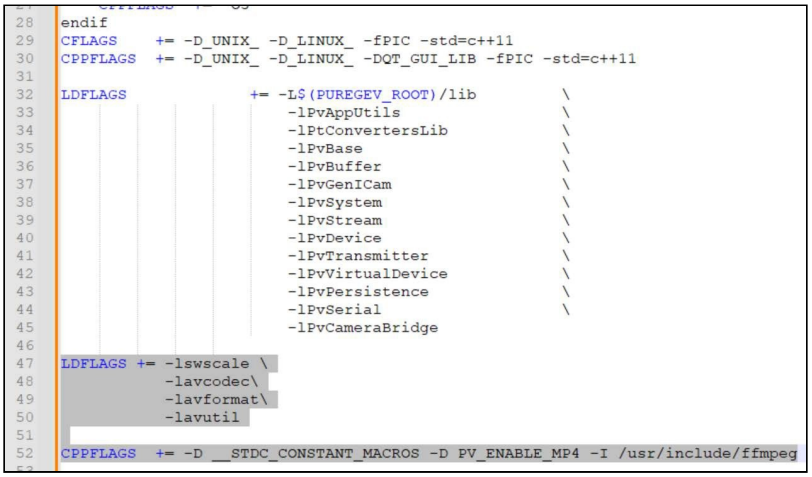
```
endif
CFLAGS    += -D_UNIX_ -D_LINUX_ -fPIC -std=c++11
CPPFLAGS  += -D_UNIX_ -D_LINUX_ -DQT_GUI_LIB -fPIC -std=c++11

LDFLAGS             += -L$(PUREGEV_ROOT)/lib          \
                        -lPvAppUtils                  \
                        -lPtConvertersLib             \
                        -lPvBase                      \
                        -lPvBuffer                    \
                        -lPvGenICam                   \
                        -lPvSystem                    \
                        -lPvStream                    \
                        -lPvDevice                    \
                        -lPvTransmitter               \
                        -lPvVirtualDevice             \
                        -lPvPersistence               \
                        -lPvSerial                    \
                        -lPvCameraBridge

LDFLAGS += -lswscale \
           -lavcodec\
           -lavformat\
           -lavutil

CPPFLAGS  += -DPV_ENABLE_MP4
```

- For RHEL 8 and CentOS 8 Stream



```
endif
CFLAGS    += -D_UNIX_ -D_LINUX_ -fPIC -std=c++11
CPPFLAGS  += -D_UNIX_ -D_LINUX_ -DQT_GUI_LIB -fPIC -std=c++11

LDFLAGS             += -L$(PUREGEV_ROOT)/lib          \
                        -lPvAppUtils                  \
                        -lPtConvertersLib             \
                        -lPvBase                      \
                        -lPvBuffer                    \
                        -lPvGenICam                   \
                        -lPvSystem                    \
                        -lPvStream                    \
                        -lPvDevice                    \
                        -lPvTransmitter               \
                        -lPvVirtualDevice             \
                        -lPvPersistence               \
                        -lPvSerial                    \
                        -lPvCameraBridge

LDFLAGS += -lswscale \
           -lavcodec\
           -lavformat\
           -lavutil

CPPFLAGS  += -D __STDC_CONSTANT_MACROS -D PV_ENABLE_MP4 -I /usr/include/ffmpeg
```

- Navigate to <your sample folder>/eBUSPlayer/
- Type “make” to recompile.

3 Installing the eBUS Runtime for Linux

eBUS Runtime Type	Requirements According to Packages
All Runtime Functionality	• build-essential for Ubuntu to build eBUS Universal Pro driver • gcc-12 for Ubuntu 22.04 to build eBUS Universal Pro driver • Qt5 is required only if the eBUS application requires display for all Linux distributions. • libavcodec5x where x is according to the Operating System. See step 1 in the “Installing the eBUS SDK for Linux”. libavcodec is required if the eBUS application requires PvGUI. • python3-numpy is required for using eBUS Python.
eBUS Edge	• build-essential for Ubuntu to build eBUS Universal Pro driver. • gcc-12 for Ubuntu 22.04 to build eBUS Universal Pro driver. • eBUS Base Runtime for all Linux distributions.
eBUS Receive	• build-essential for Ubuntu to build eBUS Universal Pro driver. • cc-12 for Ubuntu 22.04 to build eBUS Universal Pro driver. • Qt5 is required only if the eBUS application requires display for all Linux distributions. • libavcodec5x where x is according to the Operating System. See step 1 in the “Installing the eBUS SDK for Linux”. libavcodec is required if the eBUS application requires PvGUI. • eBUS Base Runtime for all Linux distributions.
eBUS Python	• python3-numpy. • python3-opencv if the eBUS Python application requires display. • eBUS Base Runtime, eBUS Receive Runtime and eBUS Edge Runtime for all distributions.

From the terminal, execute the following command/s according to the different packages.

	Packages	Commands
All Runtime Packages	eBUS Runtime	• sudo dpkg -i eBUS_Runtime_<*distribution targeted*>-<6.x.x>.deb

eBUS Edge	eBUS Base Runtime eBUS Edge Runtime	• sudo dpkg -i eBUS_Base_Runtime_<*distribution targeted*>-<6.x.x>.deb • sudo dpkg -i eBUS_Edge_Runtime_<*distribution targeted*>-<6.x.x>.deb
eBUS Receive	eBUS Base Runtime eBUS Receive Runtime	• sudo dpkg -i eBUS_Base_Runtime_<*distribution targeted*>-<6.x.x>.deb • sudo dpkg -i eBUS_Receive_Runtime_<*distribution targeted*>-<6.x.x>.deb
eBUS Python	eBUS Base Runtime eBUS Receive Runtime eBUS Edge Runtime eBUS Python Runtime	• sudo dpkg -i eBUS_Base_Runtime_<*distribution targeted*>-<6.x.x>.deb • sudo dpkg -i eBUS_Receive_Runtime_<*distribution targeted*>-<6.x.x>.deb • sudo dpkg -i eBUS_Edge_Runtime_<*distribution targeted*>-<6.x.x>.deb • sudo dpkg -i eBUS_Python_Runtime_<*distribution targeted*>-<6.x.x>.deb **OR** • sudo dpkg -i eBUS_Runtime_<*distribution targeted*>-<6.x.x>.deb

We recommend that you reboot the Linux to ensure that the correct environment variables are set.

If there is a failure to compile and install the eBUS Universal Pro driver due to permissions or missing dependencies, you can manually compile and install the driver. For more information, see “To manually install the eBUS Universal Pro for Ethernet Filter Driver” .

If the eBUS SDK installation is not successful, your system may be missing the required Linux kernel headers.
For more information, see “Error message: Cannot find the files to build kernel module in this PC".

Uninstalling the eBUS SDK (and eBUS Runtime)

You can use the dpkg command to uninstall the eBUS SDK.

To uninstall the eBUS SDK

From the terminal, execute the following command.

- For Ubuntu (x86_64 and ARM)
  - sudo dpkg -r ebus-sdk
- For RedHat/CentOS
  - sudo rpm -e ebus-sdk

To uninstall eBUS Runtime packages, Release 6.5 and higher

Execute the following commands according to the case:

- if eBUS_Python_Runtime is installed:
    - For Ubuntu (x86_64 and ARM)
        - sudo dpkg -r ebus-python-runtime
    - For RedHat/CentOS
        - sudo rpm -e ebus-python-runtime
- if eBUS_Receive_Runtime is installed:
    - For Ubuntu (x86_64 and ARM)
        - sudo dpkg -r ebus-receive-runtime
    - For RedHat/CentOS
        - sudo rpm -e ebus-receive-runtime
- if eBUS_Edge_Runtime is installed:
    - For Ubuntu (x86_64 and ARM)
        - sudo dpkg -r ebus-edge-runtime
    - For RedHat/CentOS
        - sudo rpm -e ebus-edge-runtime
- if eBUS_Base_Runtime is installed:
    - For Ubuntu (x86_64 and ARM)
        - sudo dpkg -r ebus-base-runtime
    - For RedHat/CentOS
        - sudo rpm -e ebus-base-runtime
- if eBUS_Runtime is installed:
    - For Ubuntu (x86_64 and ARM)
        - sudo dpkg -r ebus-runtime
    - For RedHat/CentOS
        - sudo rpm -e ebus--runtime

Uninstalling an old eBUS SDK (and eBUS Runtime) for Linux

1. From the terminal, execute one of the following commands.
   The command varies depending on the distribution you are using.
    - For eBUS SDK 6.3 (and higher):
        - For Ubuntu (x86_64 and ARM)
            - sudo dpkg -r ebus-sdk
        - For RedHat/CentOS
            - sudo rpm -e ebus-sdk
    - For eBUS SDK 4.0 through 6.2:
        - On Ubuntu:
            - sudo dpkg -r ebus_sdk_<*package_name*>

- Where <*package_name*> can be of the format below (dependent on the OS used and which version of eBUS SDK is installed, this list will vary, so only examples are provided):
  - Ubuntu-20.04-x86_64
  - Ubuntu-18.04-x86_64
  - Ubuntu-16.04-x86_64
- For ARM Platforms:
  - sudo dpkg -r ebus_sdk_<*package_name*>
    - linux-aarch-arm
- On RHEL/CentOS, as super user:
  - rpm -e ebus_sdk_<*package_name*>
    - rhel-centos-x86_64

To see installed packages, you can execute one of the following commands:

- On Ubuntu:
  dpkg -l | grep ebus
- On RHEL/CentOS:
  rpm -qa | grep ebus

To uninstall eBUS Runtime packages, for Release 6.4 and higher, execute the following commands:

sudo dpkg -r ebus-runtime
sudo rpm -e ebus-runtime

3.1 Choosing Optimal Settings for the Jetson Modules

We strongly recommend that you modify the Jetson Nano, AGX Xavier, Xavier NX, TX2, TX2 NX, AGX Orin and Orin NX configuration for power and performance management. If you do not perform these steps, you may encounter freezes or the eBUS SDK may stop responding.

On Jetson TX2, the MAXN mode, mode 0, is not recommended as it enables the 2 Denver cores which exhibits performance issue. Mode 3, MAXP_CORE_ARM, is a better choice on this module.

On Jetson Xavier you can select the unconstrained mode 0, MAXN, or less performing mode 3, MODE_30W_ALL

To choose the optimal nvpmodel mode

Before changing mode, it is recommended to run the following command and take note of the current mode:
nvpmodel -q

- Execute the following command:
  (**Note**: The mode is preserved on reboot.)
  sudo nvpmodel -m 3 # TX2
  sudo nvpmodel -m 0 # Xavier

To list all modes available on your Jetson module

- nvpmodel -p --verbose

To see the current clock settings

- Execute the following command:
  sudo jetson_clocks --show

To increase the CPU clock to the maximum value

The increased CPU clock speed is not preserved on reboot.

1. Back up the default value to a file by executing the following command:
   sudo jetson_clocks.sh --store <filename_to_store>
2. Increase the CPU clock speed by executing the following command:
   sudo jetson_clock

To enable jumbo frames

- The use of jumbo frames reduces the amount of Ethernet, IP, UDP, and GVSP packet overhead when transmitting images, which reduces the CPU load and memory requirements. Using the script that is detailed in “To enable jumbo packets”, set jumbo frames to **9000**.

4 Activating eBUS SDK Licenses

A license is required to take full advantage of the eBUS Edge's transmit and eBUS SDK 's receive third party GigE and USB3 vision devices. When a license is activated, the embossed watermark that appears on transmitted and received images is no longer applied, and restrictions for transmitting and receiving images are removed.

If you use the eBUS SDK without a license, the following limitations apply:

- Received images (received from third-party GigE Vision or USB3 Vision transmitter devices) have an embossed watermark.
- The raw data payload type cannot be received.
- Connections to a software-based GigE Vision device (developed with the eBUS Edge portion of the Pleora eBUS SDK) will disconnect after 15 minutes.
- Certain device information strings cannot be customized by a software developer when creating a software-based GigE Vision device, such as the device's model name.

4.1 Understanding Licensing

eBUS Receive and eBUS Edge licenses are associated to the MAC address of a network interface card (NIC). Depending on the type of license you are purchasing, you will need to provide the following information:

- For an **eBUS Receive** or **eBUS SDK Developer Seat** license, provide the MAC address of a NIC in the workstation.
- For an **eBUS Edge** license, provide the MAC address of a network interface in the embedded computer that will run your software-based GigE Vision device.

Pleora includes the MAC address in the license file that you purchase, which allows the eBUS SDK to accept the license. The MAC address is used to identify the workstation or embedded computer.

A pre-programmed USB license dongle for GigE Vision and USB3 Vision devices on Linux operating systems on x86 platforms is supported in eBUS SDK version 6.2 and later. The license dongle allows you to quickly license a system by inserting the dongle in the PC. The license dongle eliminates the need to find a MAC address and deploy a runtime license on a new system when you are using eBUS Receive, thereby minimizing system down time.
The pre-programmed USB license dongle is not supported on Linux (ARM platforms). If you need to use Linux (ARM platforms), install the license file, which is available for purchase online.

4.2 Activating an eBUS SDK License

When you activate a license on your workstation or embedded computer, the restrictions are removed.

⚠ Please take note of the following important points:
- DO NOT rename the license file provided by your Pleora
- DO NOT disable or remove the NIC (or WiFi adapter) that is associated with the

To activate an eBUS SDK license

1. On your workstation (eBUS Receive or eBUS SDK Seat license) or embedded computer (eBUS Edge license), copy the license file to:
   /opt/pleora/ebus_sdk/<*distribution_targeted*>/licenses
2. Select your next step:
   - If you are using an earlier version of eBUS SDK (version 4.0 to 6.1), go to 3.
   - If you are using eBUS SDK version 2 or later, go to 4.
3. Stop and restart the eBUS daemon by executing the following commands:
   - sudo service ebusd stop
   - sudo service ebusd restart
4. Restart any eBUS applications that are currently running (for example, eBUS Player), to ensure that the license takes effect.
   You have completed the license activation.

ⓘ **Additional Information: Stopping the eBUS daemon (applies to eBUS SDK versions 4.0 to 6.1)**
The eBUS daemon is used by the eBUS SDK for connection to USB3 Vision devices and is also used to license the eBUS SDK's transmit and receive capabilities.
With GigE Vision devices
The GigE Vision device will continue streaming images, even when the eBUS SDK daemon is not running.
Stopping the eBUS daemon has an impact on the licensing of your system. If a valid license is present when you stop the eBUS daemon, you can continue streaming images until you disconnect the GigE Vision device. At this point, an invalid license will be reported. To return to normal operation, start the eBUS daemon and restart any applications created with the eBUS SDK, such as eBUS Player or the sample applications.
With USB3 Vision devices
When you stop the eBUS daemon, any connected USB3 Vision devices will be disconnected. To connect to USB3 Vision devices with eBUS Player, the sample applications, or applications created with the eBUS SDK, the eBUS SDK must be running.

4.2.1 For More Information

For detailed information about licensing, including troubleshooting tips or using the pre-programmed USB dongle, see the *eBUS SDK Licensing Overview Knowledge Base Article*, available on the Pleora Technologies Support Center at supportcenter.pleora.com.

5 Optimizing Operation with GigE Vision Devices

This chapter provides some steps you can take to optimize operation when using GigE Vision devices with the eBUS SDK.

5.1 Using the eBUS Universal Pro for Ethernet Filter Driver

The eBUS Universal Pro for Ethernet Filter driver is automatically installed and loaded on your workstation or embedded computer during the installation of the eBUS SDK. This driver optimizes operation with GigE Vision devices. It also enhances the performance of your system by allowing GigE Vision Stream Protocol (GVSP) data to bypass some (or all) of the operating system's network stack, delivering the data directly to the eBUS SDK.

When installing the eBUS SDK, if the eBUS Universal Pro for Ethernet Filter Driver fails to compile and install the eBUS Universal Pro driver due to permissions or missing dependencies, you can manually compile and install the driver.

If you would like to uninstall the filter driver, you can do so. Keep in mind that you can still work with GigE Vision devices after you uninstall the eBUS Universal Pro for Ethernet Filter driver. However, the operation of these devices is not optimized.

To manually install the eBUS Universal Pro for Ethernet Filter Driver

1. Navigate to the following directory:
   /opt/pleora/ebus_sdk/<*distribution_targeted*>/module
2. Execute the following command:
   sudo ./build.sh --kernel=/usr/src/<*linux_headers*>
   Where <*linux_headers*> is the corresponding headers for the Linux4Tegra version that is installed. For example, **linux-headers-4.4.38-tegra**.

   If you are using JetPack 4.6.3, navigate to the following directory:
   /opt/pleora/ebus_sdk/linux-aarch64-arm/module/
   Execute the following command:
   sudo ./build.sh --kernel=/usr/src/linux-headers-4.9.299-tegra- ubuntu18.04_aarch64/kernel-4.9/

   If you are using JetPack 5.1, navigate to the following directory:
   /opt/pleora/ebus_sdk/linux-aarch64-arm/module/
   Execute the following command:
   sudo ./build.sh --kernel=/usr/src/linux-headers-5.10.104-tegra- ubuntu20.04_aarch64/kernel-5.10
3. After you compile the driver, install it by executing the following command:
   sudo ./install_driver.sh --install

To uninstall the eBUS Universal Pro for Ethernet Filter Driver

- Execute the following script and follow the prompts:
  sudo /opt/pleora/ebus_sdk/<*distribution_targeted*>/module/install_driver.sh --uninstall

To start, stop, or check the status of the eBUS Universal Pro for Ethernet Filter Driver

- Execute the following command:
  sudo /opt/pleora/ebus_sdk/<*distribution_targeted*>/module/ebdriverlauncher.sh <*command*> (where <*command*> can be start, stop, or status)

> You can also check the status of the driver by executing the following command:
> lsmod | grep ebUniversalProForEthernet

5.2 Enabling Jumbo Ethernet Frames

If supported by your network card, GigE Vision device, and switch (if applicable), we recommend that you enable jumbo Ethernet frames when using GigE Vision devices. The use of jumbo frames reduces the amount of Ethernet, IP, UDP, and GVSP packet overhead when transmitting images, which reduces the CPU load and memory requirements.

To enable jumbo packets

- Execute the following command:
  sudo ifconfig <*network_interface_ID*> mtu [*SIZE*]
  where, [*network_interface_ID*] is the name of the network interface card (NIC) [*SIZE*] = desired frame size
  For example:
  sudo ifconfig eth0 mtu 9000

> To see the name of the NIC in your system, execute the following command:
> sudo lshw -c network -businfo
> To see the current maximum transmission unit (MTU) value, execute the following command:
> ifconfig | grep MTU

5.3 Additional optimization steps

For GigE Vision devices, when a driver is installed, further optimization can be achieved by running the set_socket_buffer_size.sh script located in /opt/pleora/ebus_sdk/ <*distribution_targeted*>/bin, as sudo with the following command:

sudo ./set_socket_buffer_size.sh.

For further information, see the *eBUS SDK C++ API Quick Start Guide* or the e*BUS SDK Python API Quick Start Guide* for PvStreamGEV::SetUserModeSocketRxBufferSize details for Linux virtual NIC support.

6 Providing Access to Third-Party USB3 Vision Transmitter Devices

To access third-party, non-Pleora USB3 Vision transmitter devices, you must add the device's vendor ID to the eBUS SDK.

💡 If you are not sure if your device is a Pleora transmitter, observe the USB GUID that appears on the device's label or in your software application. If it begins with the Pleora vendor ID (28b7), it uses Pleora's transmitter technology.

To set up access for USB3 Vision devices that are not enabled with Pleora's technology

1. Execute the following command (this command works on any Linux platform): ./opt/pleora/ebus_sdk/<*distribution_targeted*>/bin/set_udev_rules.sh
2. When prompted, enter the vendor ID assigned to the device's This number often appears on the device's label.
3. For eBUS versions 0 to 6.1, stop and restart the eBUS daemon by executing the following commands: sudo service eBUSd restart

7 Using eBUS Player to Configure Devices and Stream Images

You can use eBUS Player (which is precompiled and installed with the release in the bin folder) to connect to, configure, and stream images from GigE Vision and USB3 Vision devices.

You must disable the firewall before using GigE Vision devices. Some optimization may be desirable before using GigE Vision or USB3 Vision devices with the eBUS SDK in a production environment, as was discussed in “Optimizing Operation with GigE Vision Devices”.

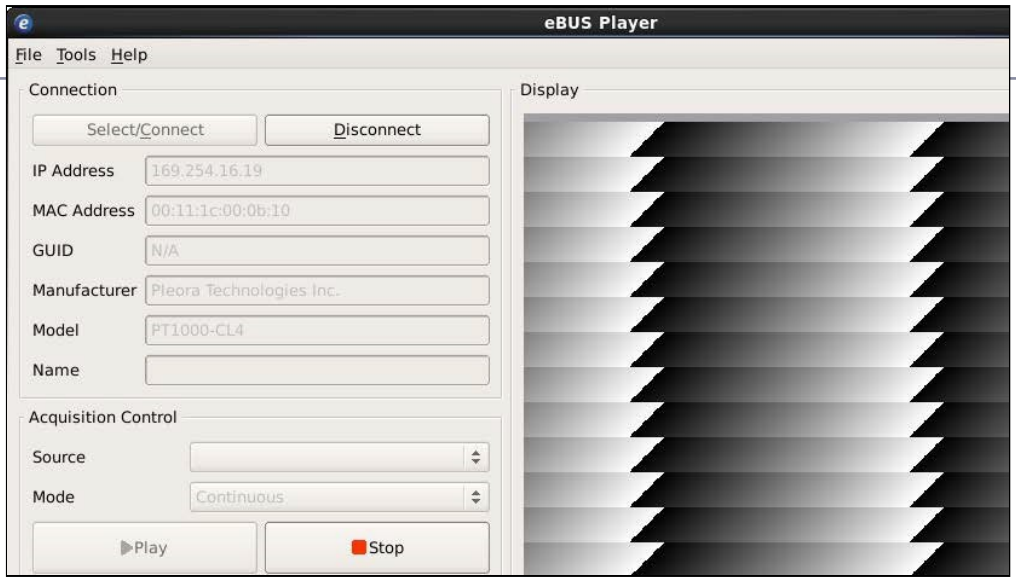
Qt is required to run eBUS Player. Relevant Qt version must be installed for your particular OS prior to launching/running eBUS Player.

When starting eBUS Player on Ubuntu 22.04 or CentOS 8 Stream or RedHat platforms, the following warning is displayed in the terminal:
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland to run on Wayland anyway.

Workaround for **Ubuntu 22.04**, at the login screen, before entering your password, locate the cog wheel in the lower right corner of the screen, and click on it. Select **Ubuntu on xorg** and proceed with entering your password.

Workaround for **CentOS 8 Stream** and **RedHat**, at the login screen, before entering your password, locate the cog wheel in the lower right corner of the screen, and select a session with **x11 display server** and proceed with entering your password.

In the opt/pleora/*<distribution_targeted>*/ebus_sdk/bin folder, you will find both an eBUSPlayer script and an eBUSPlayer.bin executable. You should always execute the eBUSPlayer script, which sets the proper environment variables, and then automatically executes the eBUSPlayer.bin executable.


eBUS Player
File Tools Help
Connection
Select/Connect
Disconnect
IP Address
169.254.16.19
MAC Address
00:11:1c:00:0b:10
GUID
N/A
Manufacturer
Pleora Technologies Inc.
Model
PT1000-CL4
Name
Acquisition Control
Source
Mode
Continuous
Play
Stop
Display

8 Compiling and Running Sample Applications

We strongly recommend that you copy the sample applications from the /opt/pleora/ebus_sdk/<distribution_targeted>/share/ directory to your personal development workspace before modifying or compiling them.

A list of C++ samples, with a description of each, can be found in the index.html file in the /opt/ pleora/ebus_sdk/<distribution_targeted>/share/samples directory.

Compilation will fail for GUI-based samples if the Qt development package is not installed. For Qt version information, see "System Requirements".

A list of Python samples can be found in the /opt/pleora/ebus_sdk/<distribution_targeted>/ share/samples/python/ebus directory.

For information about building the sample applications and creating your own applications, see the *eBUS SDK C++ API Quick Start Guide*, and the *eBUS SDK Python API Quick Start Guide*.

To compile the eBUS SDK C++ sample applications

1. If you have not restarted your computer since installing the eBUS SDK, we recommend that you do so This ensures that the correct environment variables are set at startup.
2. Make a copy of the following directory, as a backup for the original source files: /opt/pleora/ebus_sdk/<*distribution_targeted*>/share/samples
3. Navigate to the directory that contains the copy of the sample code (that was created in Step 2 above).
4. Do one of the following:
   - **To compile all sample applications at one time**: Execute the sh script by typing ./build.sh
   - **To compile a specific sample application**: Navigate to the directory of the sample (for example, MulticastMaster) and execute the make command.

To use the eBUS SDK Python sample applications

1. If you have not restarted your computer since installing the eBUS SDK, we recommend that you do so This ensures that the correct environment variables are set at startup.
2. Make a copy of the following directory, as a backup for the original source files: /opt/pleora/ebus_sdk/<*distribution_targeted*>/share/samples/python/ebus
3. Navigate to the directory that contains the copy of the sample code (that was created in Step 2).
4. Launch the Python application by executing the following command:
   - **python3 ./<python_sample>.py**.

If you encounter issues with running the samples due to the environment variables not being set, you can restart your computer or set the environment variables manually, as outlined in “The sample applications compiled successfully, but they will not run.”.

9 Troubleshooting

Cannot detect or connect to GigE Vision devices.

In the **Device Selection** dialog box, select the **Show unreachable Network Devices** check box. If the device still does not appear, do one of the following:

- Disable the firewall.
  - And/or -
- Execute the following script and then choose option **0** (to disable strict Reverse Path Forwarding filtering):
  sudo /opt/pleora/ebus_sdk/<*distribution_targeted*>/bin/set_rp_filter.sh

Cannot detect or connect to USB3 Vision devices.

For eBUS SDK versions 4.0 to 6.1, ensure that the eBUS daemon is started by executing the command: sudo service eBUSd status

- If the daemon is not started, execute the command: sudo service eBUSd start
- If the daemon is not installed, run the sudo /opt/pleora/ebus_sdk/<distribution_targeted>/ bin/ install_daemon.sh script and follow the

If the device does not use Pleora's transmitter technology and the device's vendor ID has not been added to the eBUS SDK, you will be unable to detect or connect to it. To see if your device uses Pleora transmitter technology, observe the USB GUID that appears on the device's label or in your software application. If it begins with the Pleora vendor ID (28b7), it uses Pleora transmitter technology. For information about accessing the camera, see "Providing Access to Third-Party USB3 Vision Transmitter Devices" .

A high number of GTK errors or warnings appear when running an application.

It is likely that you are running the sample application as superuser, but are logged on as a standard user.

The sample applications compiled successfully, but they will not run.

As superuser, execute the /opt/pleora/ebus_sdk/<*distribution_targeted*>/bin/install_libraries.sh script to ensure that the required libraries have been added to the system. Also, ensure that you source the bin/ set_puregev_env.sh script to set the environment variables (source set_puregev_env.sh).

You can also launch the sample applications using the RunFromEnv.sh script. For example: source ./RunFromEnv.sh && ~/samples/PvStreamSample/PvStreamSample

In this example, it is assumed that a copy of the sample applications is available in your Home directory and that you have "write" access.

The following error appears:GENICAM_ROOT_V3_4 is not set.

Restart the computer. This issue can occur if you did not restart the computer after installing the eBUS SDK. If the issue persists, follow the steps to set the environment variables, as outlined above in “The sample applications compiled successfully, but they will not run.”.

A watermark appears on received images.

You require a receive license. For more information, see “Activating eBUS SDK Licenses” or the *eBUS SDK Licensing Overview Knowledge Base Article*, available on the Pleora Support Center (supportcenter.pleora.com).

Error message: Cannot find the files to build kernel module in this PC

This error, which can occur when you are installing the eBUS SDK or eBUS Universal Pro for Ethernet Filter Driver, indicates that the kernel headers are not present on your system. The kernel headers are required to compile one of the Linux modules and they must match the system kernel version.

To install the Linux kernel headers, execute one of the following commands:

- On Ubuntu:
  sudo apt-get install linux-headers-$(uname -r)
- On RHEL/CentOS, as super user:
  yum install kernel-devel

The eBUS SDK applications freeze or stop responding on the Jetson module

Ensure that you have chosen the nvpmodel mode, increased the CPU clock the maximum value, and configured jumbo packets for 9000 bytes. For more information, see “Choosing Optimal Settings for the Jetson Modules” .

The eBUSd daemon is not loaded and cannot restart (in eBUS SDK versions 4.0 to 6.1)

If you are using eBUS SDK versions 4.0 to 6.1 and want to activate a license on the Linux operating system, but the eBUSd is failing to restart and is reported as “not loaded”, for example:
sudo service eBUSd stop
Failed to stop eBUSd.service: Unit eBUSd.service not loaded
Ensure that the eBUS daemon is installed and configured to automatically load at startup:
cd /opt/pleora/ebus_sdk/<Distribution_Targeted_by_eBUS_installer>/bin sudo ./install_daemon.sh

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.
- Most material is available without logging in to a Support Center account.
- To access software and firmware downloads, in addition to other content, log in to the Support Center.
- If you do not have an account, click Request Account.
- Accounts are usually validated within one business day.

Copyright Information

Document ID: QSG-0011