1. Abstract

This guide explains how to get started with the MicroVision Linux Software Development Kit (SDK) for its 4th Generation PicoP Scanning Engines (PSE). The PicoP Linux SDK allows developers to quickly and easily integrate PicoP Scanning Engine control into a Linux application. The Linux Application can communicate with PSE over USB. The SDK package includes the C Application Programming Interface (API), SDK libraries, documentation, and sample projects that demonstrate use of some basic (PSE) functions. The sample applications have been developed and built using GNU command line tools.

2. Table of Contents

3. Installing PicoP SDK

The latest PicoP Linux SDK can be cloned or downloaded from:

https://github.com/MicroVision-Inc/Interactive_Projection

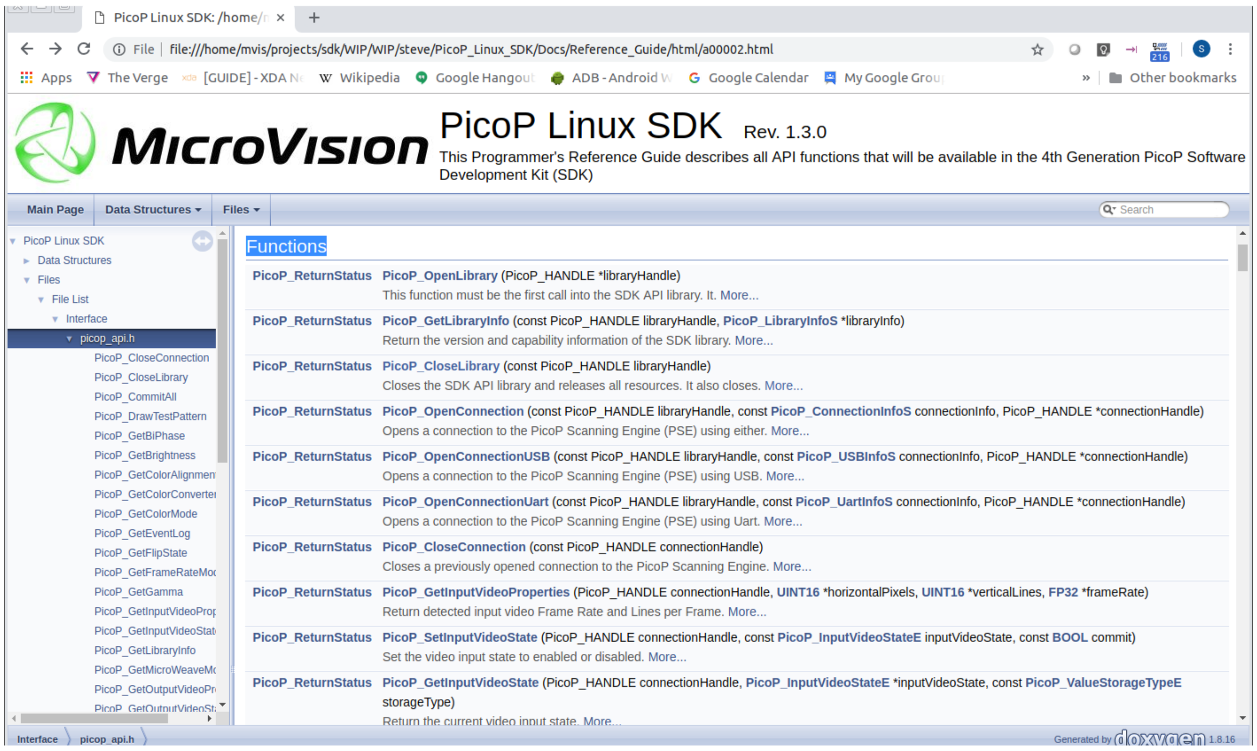
Alternatively, the SDK may be delivered as a compressed file named *PicoP_Linux_SDK_Ver_X_Y_Z.zip*. The X_Y_Z postfix of the file name represents the version number of the SDK (X equals the major version, Y the minor version, and Z the patch number of the SDK). To install the SDK, simply decompress the file into *a* directory of your choice. The cloned or uncompressed destination folder will contain the following subfolders:

PicoP_Linux_SDK				*SDK Root folder*
	Docs			*SDK Documentation*
		Reference_Guide/html		*Detailed Programmer's Reference Guide*
			index.html	*Programmer's Reference Guide entry point*
			*.html	*Programmer's Reference Guide documentation*
		DA0140028_PicoP_Gen4_Programmers_Guide.pdf		*High-Level Programmer's Guide*
		EULA.pdf		*SDK End User License Agreement*
	Interface			*PicoP SDK API Interface*
		picop_api.h		*PicoP SDK header file with function prototypes*
		picop_def.h		*PicoP SDK definitions files*
		picop_rc.h		*PicoP SDK return codes*
	Lib			*PicoP SDK Linux libraries (64-bit)*
		libPicoP_Api.so.1.3.0		*PicoP SDK shared library*
		libPicoP_Api.a		*PicoP SDK static library*
	Samples			*SDK Sample Applications*
		Bin		*Contains the 64-bit executables for the sample applications.*
		PicoP_Console		*Simplest Console application demonstrating how to communicate with PSE*
		PicoP_Display		*Simple GUI Application demonstrating the basic usage of sample C APIs for Display functions*
		PicoP_Update_FW		*Simplest Console application demonstrating how to perform a FW update on the PSE*
	DA0140038_PicoP_Gen4_Linux_SDK_Getting_Started_Guide.pdf			*This Getting Started guide for PicoP Linux SDK*

4. Accessing SDK Documentation

For high level description of functions/commands supported by the PSE Application Programming Interface (API), please refer to the *DA0140028_PicoP_Gen4_Programmers_Guide.pdf* .

For detailed description of the C-language API, please refer to the Programmer's Reference Guide at *Docs\Reference_Guide\index.html*. The Reference Guide is a set of hyperlinked HTML files containing detailed description of all Function interfaces and definitions provided by the API.



PicoP Linux SDK Rev. 1.3.0

This Programmer's Reference Guide describes all API functions that will be available in the 4th Generation PicoP Software Development Kit (SDK)

Functions

PicoP_ReturnStatus	PicoP_OpenLibrary (PicoP_HANDLE *libraryHandle) This function must be the first call into the SDK API library. It. More...
PicoP_ReturnStatus	PicoP_GetLibraryInfo (const PicoP_HANDLE libraryHandle, PicoP_LibraryInfoS *libraryInfo) Return the version and capability information of the SDK library. More...
PicoP_ReturnStatus	PicoP_CloseLibrary (const PicoP_HANDLE libraryHandle) Closes the SDK API library and releases all resources. It also closes. More...
PicoP_ReturnStatus	PicoP_OpenConnection (const PicoP_HANDLE libraryHandle, const PicoP_ConnectionInfoS connectionInfo, PicoP_HANDLE *connectionHandle) Opens a connection to the PicoP Scanning Engine (PSE) using either. More...
PicoP_ReturnStatus	PicoP_OpenConnectionUSB (const PicoP_HANDLE libraryHandle, const PicoP_USBInfoS connectionInfo, PicoP_HANDLE *connectionHandle) Opens a connection to the PicoP Scanning Engine (PSE) using USB. More...
PicoP_ReturnStatus	PicoP_OpenConnectionUart (const PicoP_HANDLE libraryHandle, const PicoP_UartInfoS connectionInfo, PicoP_HANDLE *connectionHandle) Opens a connection to the PicoP Scanning Engine (PSE) using Uart. More...
PicoP_ReturnStatus	PicoP_CloseConnection (const PicoP_HANDLE connectionHandle) Closes a previously opened connection to the PicoP Scanning Engine. More...
PicoP_ReturnStatus	PicoP_GetInputVideoProperties (PicoP_HANDLE connectionHandle, UINT16 *horizontalPixels, UINT16 *verticalLines, FP32 *frameRate) Return detected input video Frame Rate and Lines per Frame. More...
PicoP_ReturnStatus	PicoP_SetInputVideoState (PicoP_HANDLE connectionHandle, const PicoP_InputVideoStateE inputVideoState, const BOOL commit) Set the video input state to enabled or disabled. More...
PicoP_ReturnStatus	PicoP_GetInputVideoState (PicoP_HANDLE connectionHandle, PicoP_InputVideoStateE *inputVideoState, const PicoP_ValueStorageTypeE storageType) Return the current video input state. More...

5. Using the PicoP Application Programming Interface (API)

The PicoP SDK for Linux allows you to easily interface your application software with the PicoP Scanning Engine. This section explains the steps of the integration process.

5.1. SDK Version Control

The SDK API version can be queried with the **`PicoP_GetLibraryInfo()`** function. The Library info returns the major (X), minor (Y), and patch number (Z) for the SDK Version X.Y.Z. The library information also includes capability flags which can differentiate library implementation enhancements that are compatible with the same API and header files.

```
// Library information
typedef struct {
   UINT8  majorVersion;    // Contains the major version of the library
   UINT8  minorVersion;    // Contains the minor version of the library
   UINT8  patchVersion;    // Contains the patch version of the library
   UINT32 capabilityFlags; // Flags that describe the capability of the library
} PicoP_LibraryInfoS;
```

5.2. Step 1: Link SDK Library to the application

To include the PicoP Linux SDK to an application, include the **`picop_api.h`** header file into the application source code and link the **`libPicoP_Api.a`** static library or **`libPicoP_Api.so.1.3.0`** shared library to the application executable.

5.3. Step 2: Initialize the SDK Library

The first step in connecting to a PicoP is to initialize the PicoP library by calling **`PicoP_OpenLibrary()`**. The **`PicoP_OpenLibrary()`** function returns a handle to the library that can be used to open a connections to the PicoP device.

5.4. Step 3: Connect to PicoP

After successful initialization of the library, the next step is to create a connection to the PicoP device. The connection can be established using either Universal Serial Bus (USB) or Universal Asynchronous Receiver-Transmitter (UART) physical interfaces. To create a connection to the PicoP Scanning Engine, call either the **`PicoP_OpenConnection()`**, **`PicoP_OpenConnectionUSB()`**, `or` **`PicoP_OpenConnectionUart()`** functions. Upon successful connection, the library will return a connection handle to be used with subsequent library calls. The connection handle identifies the connected system.

5.5. Step 4: Configure and Control PicoP

The PicoP API is split into the following functional categories:

- **Connection & Library Management**: Connect to PicoP over USB or Serial (UART) interface.
- **Display Control**: Configure and control the PicoP Display.
- **System Management**: Manage the PicoP System, Firmware upgrades, Event Log, etc.
- **Input Control**: Configure the PicoP Input Video.
- **3D Sensing:** Configure and control PicoP 3D Sensing interface and retrieve 3D point cloud data.
- **Rendering**: Render images and test patterns into Framebuffer and On-Screen Display (OSD).

5.5.1. Connection Management

The Connection Management Functions are used to connect to the PicoP Display Engine using USB or UART.

Function	Description
PicoP_OpenLibrary( )	Opens the library and allocates resources necessary for operation. It returns a handle to the library that must be used in subsequent calls
PicoP_GetLibraryInfo( )	Retrieves the version and capability information of the SDK Library
PicoP_CloseLibrary( )	Closes the library and releases all resources. It also closes all the open connections.
PicoP_OpenConnection( )	Opens a connection to the PicoP Display Engine using either USB or UART.
PicoP_CloseConnection( )	Closes a previously opened connection to the PicoP Display Engine.

5.5.2. Display Control Functions

The Display Control Functions can be used to configure the output display.

Function	Description
PicoP_SetBrightness( )	Sets brightness for the output display.
PicoP_GetBrightness( )	Returns brightness setting of the output display.
PicoP_SetColorMode( )	Sets color mode for the output display.
PicoP_GetColorMode( )	Returns color mode setting of the output display.
PicoP_SetGamma( )	Sets gamma value for the output display.
PicoP_GetGamma( )	Returns gamma value setting of the output display.
PicoP_SetVideoGammaBoostMode( )	Set the color specific video gamma boost mode.
PicoP_GetVideoGammaBoostMode( )	Return the color specific video gamma boost mode.
PicoP_SetMicroWeaveMode( )	Select the MicroWeave mode.
PicoP_GetMicroWeaveMode( )	Return the MicroWeave mode.
PicoP_SetFlipState()	Sets the flip state of the image to horizontal, vertical, both horizontal and vertical or none.
PicoP_GetFlipState()	Returns the current flip state of the output display.
PicoP_SetOutputVideoState( )	Set the video output state to enabled or disabled.
PicoP_GetOutputVideoState( )	Returns the current video output state.
PicoP_GetOutputVideoProperties( )	Return output video properties.
PicoP_SetBiPhase( )	Sets the scan line phase delay to align the forward and reverse scan video.
PicoP_GetBiPhase( )	Returns the scan line phase delay setting.
PicoP_SetColorAlignment()	Performs vertical or horizontal color alignment for the selected color
PicoP_GetColorAlignment()	Gets the color alignment offset of the chosen color

PicoP_SetColorConverter()	Sets the color converter values
PicoP_GetColorConverter()	Gets the color converter values
PicoP_SetFrameRateMode( )	Sets the output frame rate mode (frame rate vs. vertical display and sensing resolutions) and display scaling.
PicoP_GetFrameRateMode( )	Returns the output frame rate mode (frame rate vs. vertical display and sensing resolutions) and display scaling

5.5.3. System Management Functions

The System Management Functions are used to control the PicoP System and to access the system information.

Function	Description
PicoP_GetSystemInfo( )	Retrieves system information.
PicoP_GetSystemStatus( )	Retrieves the system status.
PicoP_GetEventLog( )	Retrieves the system event log.
PicoP_RestoreFactoryConfig( )	Restores System Settings to Factory Configuration.
PicoP_CommitAll( )	Commits all user settings to flash.
PicoP_UpgradeSoftware( )	Upgrades the embedded Software.

5.5.4. Input Control Functions

The Input Control functions are used to configure the PicoP Input Video.

Function	Description
PicoP_GetInputVideoProperties( )	Returns detected input video Frame Rate and Lines per Frame.
PicoP_SetInputVideoState( )	Enables or Disables the input video. When input video is disabled, the framebuffer will not be updated and the output video will contain the last captured frame.
PicoP_GetInputVideoState( )	Returns the current state of the input video.

5.5.5. 3D Sensing Functions

The 3D Sensing functions are used to configure the 3D sensor.

Function	Description
PicoP_SetSensingState( )	Turns the 3D Sensing function on or off. When the 3D Sensing function is off, the IR laser is not pulsing and no 3D sensing data is sent over the USB.
PicoP_GetSensingState( )	Returns the current Sensing state.

5.5.6. Rendering Functions

The Rendering function allows the host system to render information into the PicoP On-Screen Display (OSD) or FrameBuffer.

Function	Description
PicoP_DrawTestPattern( )	Displays one of the built-in test patterns.

5.6. *Step 5: Exit Application*

To gracefully exit the host application, call the **`PicoP_CloseConnection()`** and **`PicoP_CloseLibrary()`** functions to shut down the connection to PicoP and to release all resources used by the library.

6. SDK Host Platforms

The PicoP Gen4 SDK and applications has been verified on the following hardware platforms ("$ cat /etc/os-release", "$ uname -r"):

- PC : Ubuntu Linux 16.04
    - DISTRIB_ID=Ubuntu
    - DISTRIB_RELEASE=16.04
    - DISTRIB_CODENAME=xenial
    - DISTRIB_DESCRIPTION="Ubuntu 16.04.5 LTS"
    - NAME="Ubuntu"
    - VERSION="16.04.5 LTS (Xenial Xerus)"
    - ID=ubuntu
    - ID_LIKE=debian
    - PRETTY_NAME="Ubuntu 16.04.5 LTS"
    - VERSION_ID="16.04"
    - HOME_URL="http://www.ubuntu.com/"
    - SUPPORT_URL="http://help.ubuntu.com/"
    - BUG_REPORT_URL="http://bugs.launchpad.net/ubuntu/"
    - VERSION_CODENAME=xenial
    - UBUNTU_CODENAME=xenial
    - kernel version: 4.15.0-36-generic

6.1. *Required Platform Packages*

Each Linux platform requires the installation of the following packages:

- libudev-dev, (GNU Lesser General Public License (LGPL), version 2.1)
- libbluetooth-dev, (GNU General Public License (GPL))
- libusb-1.0-0-dev, (GNU Lesser General Public License (LGPL), version 2.1)
- v4l-utils, (GNU Lesser General Public License)
- build-essential
- git

7. Demo Applications

PicoP_Linux_SDK/samples/ contains the following demo applications utilizing the PicoP SDK:

- PicoP_Console
- PicoP_Display
- PicoP_Update_FW

To build any of these applications, open a shell and navigate to the application directory and execute "make". This will build the application using the static library.

There is no Advanced Packaging Tool (APT) package created for shared library installation. To evaluate the shared library, softlink creation is required to traverse the versions of the soname. For example, if released version is libPicoP_ALC_Api.so.1.0.0 then create following softlinks in library path directory:

```
$ sudo ln -sf libPicoP_Api.so.1.3.0 libPicoP_Api.so.1.3
$ sudo ln -sf libPicoP_Api.so.1.3 libPicoP_Api.so.1
$ sudo ln -sf libPicoP_Api.so.1 libPicoP_Api.so
```

The library utilizes USB device node access for communication and video streaming. This requires running all demo applications under sudo access control or create udev rules to allow user space programs access to usb device nodes. There are several ways to create the udev rules, but one simple rule set which allows device node access to any user on the host is the following 2 lines:

- SUBSYSTEMS=="usb", ENV{DEVTYPE}=="usb_device", ATTRS{idVendor}=="148a", ATTRS{idProduct}=="0004", MODE="0666"
- SUBSYSTEMS=="usb", ENV{DEVTYPE}=="usb_device", ATTRS{idVendor}=="04b4", ATTRS{idProduct}=="00f9", MODE="0666"

Add these 2 lines to a udev file (filename extension .rules) located in /etc/udev/rules.d. Note if your chosen filename exists in /lib/udev/rules.d directory those rules will be overridden, so it is advisable to choose a unique filename for these udev rules.

7.1. PicoP_Console Sample Project

The **`PicoP_Console`** sample project is a simple Linux console application. It demonstrates the use of SDK interfaces to communicate with the PicoP device and to control basic Projector functions. The application can control the following PicoP operations:

- Retrieve library info and PSE system status
- Retrieve/Change display brightness setting

- Retrieve/Change display color mode setting
- Flip display
- Retrieve System Status
- Retrieve/Change Horizontal Scan Angle
- Retrieve/Change Vertical Scan Angle
- Present Test Patterns

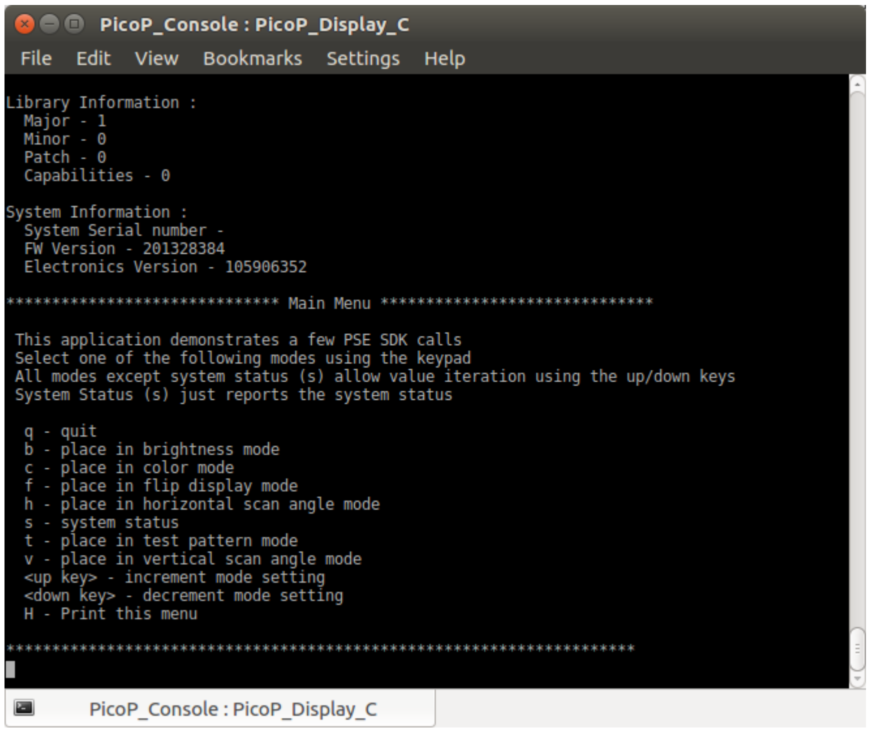
To build the application using a static library invoke make with no arguments:

```
$ make
```

To build using a shared library, configure softlinks as detailed above, and invoke make with the STATIC argument:

```
$ make STATIC=false
```

To run the application issue “sudo ./PicoP_Display_Console_Demo” at the shell command line to run the application. As shown below, Library and System information will be displayed in the shell followed by help instructions. Select one of the supported modes, and then press the up key (key value 65) or down key (key value 66) to cycle though configurations for that mode.


PicoP_Console : PicoP_Display_C
File Edit View Bookmarks Settings Help
PicoP_Console : PicoP_Display_C

```
Library Information :
  Major - 1
  Minor - 0
  Patch - 0
  Capabilities - 0

System Information :
  System Serial number -
  FW Version - 201328384
  Electronics Version - 105906352

****************************** Main Menu ******************************

 This application demonstrates a few PSE SDK calls
 Select one of the following modes using the keypad
 All modes except system status (s) allow value iteration using the up/down keys
 System Status (s) just reports the system status

  q - quit
  b - place in brightness mode
  c - place in color mode
  f - place in flip display mode
  h - place in horizontal scan angle mode
  s - system status
  t - place in test pattern mode
  v - place in vertical scan angle mode
  <up key> - increment mode setting
  <down key> - decrement mode setting
  H - Print this menu

*********************************************************************
```

7.2. PicoP_Display Sample Project

The PicoP_Display_GUI_Demo application presents a GUI to exercise several more of the Display control API functions. The application can control the following PicoP operations:

- Retrieve library info and PSE system status
- Retrieve/Change display color mode setting
- Retrieve/Change Input Video State
- Retrieve/Change Output Video State
- Retrieve Output Video Properties
- Retrieve/Change Frame Rate Mode
- Retrieve/Change Brightness value
- Retrieve/Change Brightness Compensation Mode
- Retrieve/Change Color Mode
- Retrieve/Change Color Conversion
- Retrieve/Change Gamma
- Retrieve/Change Gamma Boost Mode
- Retrieve/Change Microweave Mode
- Retrieve/Change BiPhase value
- Change Flip State
- Perform Factory Configuration Reset
- Present Test Patterns
- Perform Commit All and Factory Reset

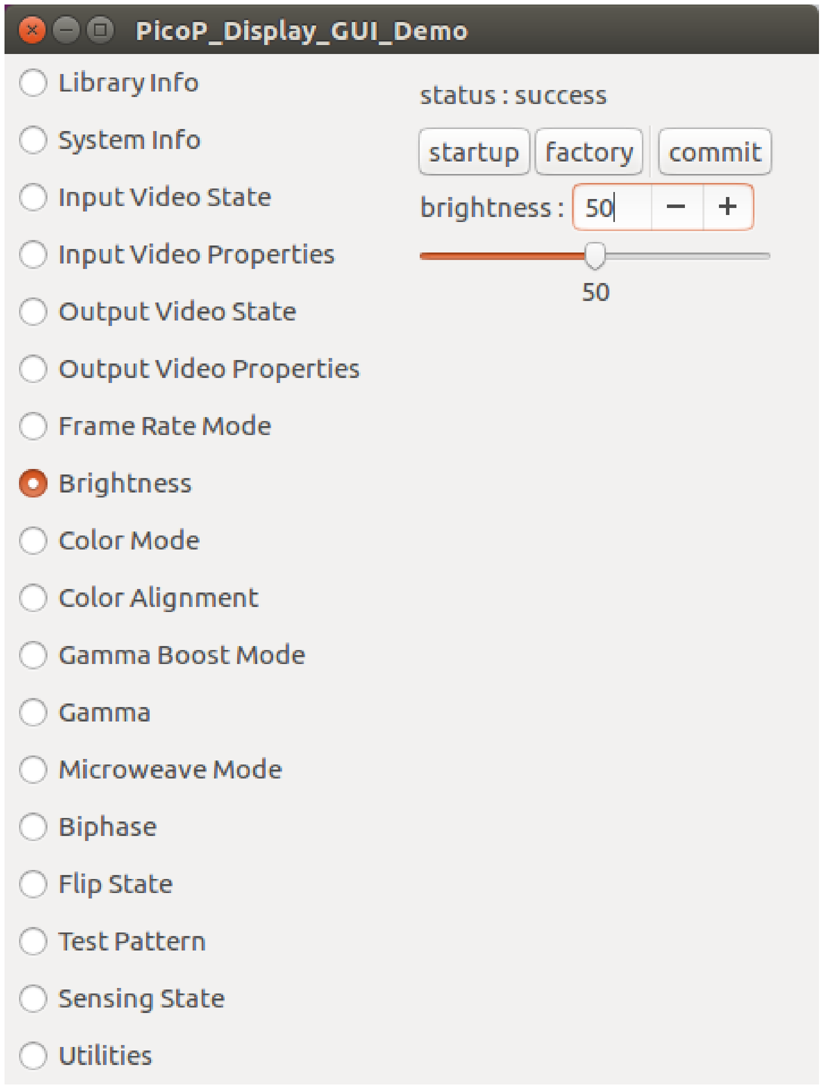
Installation of the following platform package is required for this application:

- libgtkmm-3.0-0-dev, (version >= 3.18.0-1) (GNU Lesser General Public License)

To build the application using a static library invoke make with no arguments:

```
$ make
```

To run the application issue "sudo ./PicoP_Display_GUI_Demo" at the shell command line to run the application. As shown below, a GUI application launches with supported commands enumerated on the left side of the pane. Selecting a command mode radio button will show its control widgets on the right side of the pane.



status : success

brightness : 50

50

7.3. *PicoP_Update_FW Sample Project*

The PicoP_Update_FW is a command line application for performing a FW update. Place the FW binary named "pde_client.chunk.bin" in the same directory as the PicoP_Update_FW_Demo executable. Connect USB to the Dev Kit target and run the executable. The console will display the FW update progress.