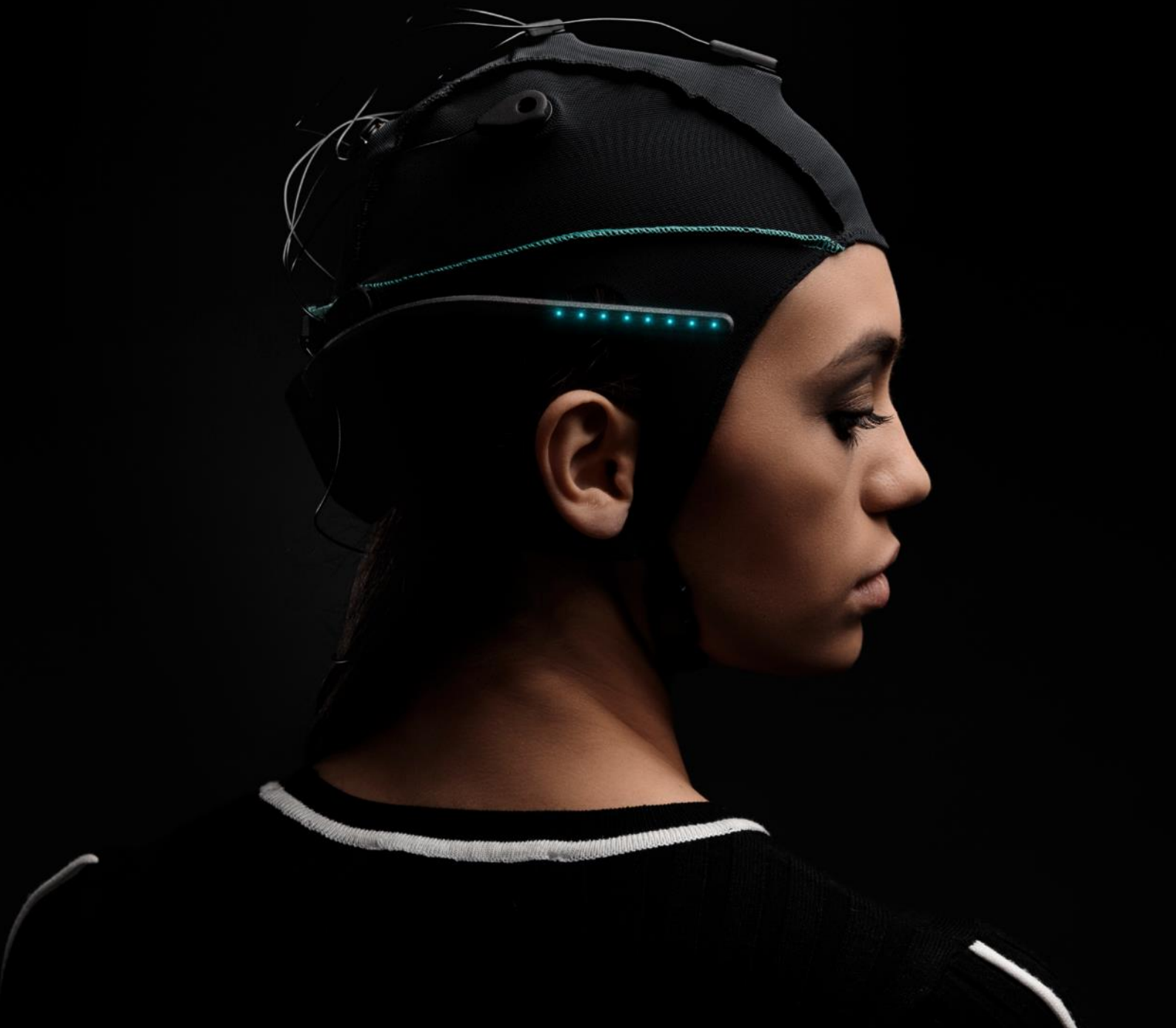
LINUX C API

USER MANUAL



User Manual for Unicorn Linux C API

Version Name: Unicorn Hybrid Black

Version Number: 1.18.00

www.unicorn-bi.com | hello@unicorn-bi.com

CONTENTS

1. UNICORN LINUX C API 1

1.1. Requirements 1

1.2. Library Files 1

1.3. Setting up a project using Eclipse 2

1.4. Command Order 7

1.5. Unicorn C API – C Reference 8

1.5.1. Constants 8

1.5.2. Error Codes 9

1.5.3. Type definitions 10

1.5.4. Structures 11

1.5.5. Functions 13

1. UNICORN LINUX C API

The Unicorn Linux C API is a C/C++ application programming interface (API) enabling the communication with Unicorn brain interfaces from C/C++ applications on Linux machines. The Unicorn Linux C API allows users to acquire data from Unicorn Brain Interfaces easily without having to take care of low-level data acquisition issues. The raw binary data stream is converted into numerical values such that the user receives data ready to analyze.

1.1. REQUIREMENTS

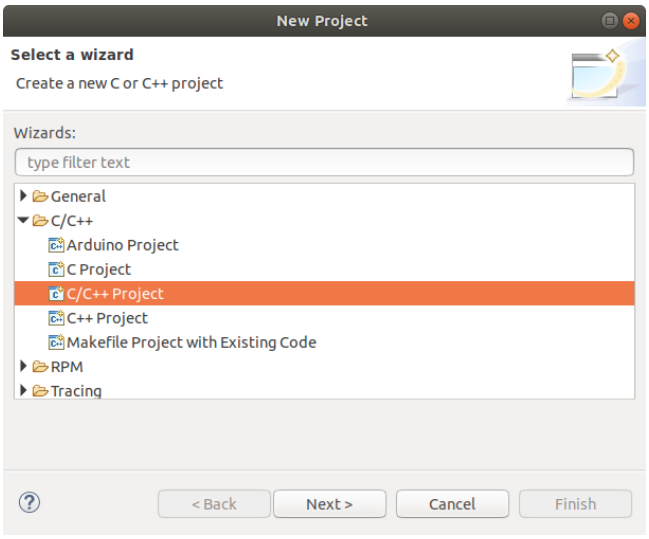
Software	Properties
Eclipse	Eclipse IDE for C/C++ Developers 2019-09 (4.13.0)
Linux	Ubuntu 18.04 64 bit
Bluez	Bluez 5.48

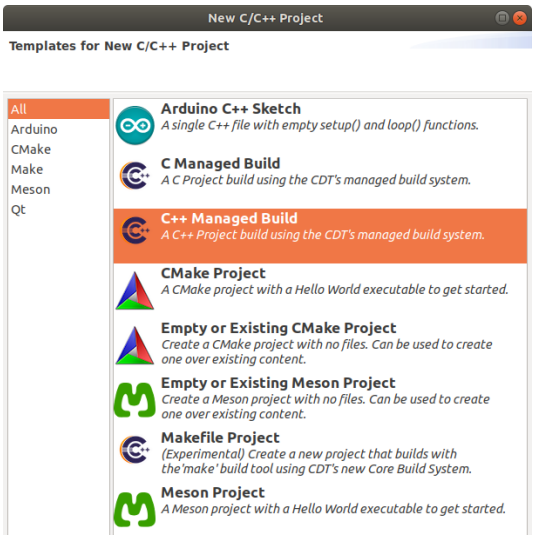
1.2. LIBRARY FILES

By default, the Unicorn Linux C API library consists of the following directories and files	
.\Lib	Contains the Unicorn Linux C API for Linux 64-bit
.\UnicornCAPIAcquisitionExample	Contains a data acquisition example for the Unicorn Linux C API

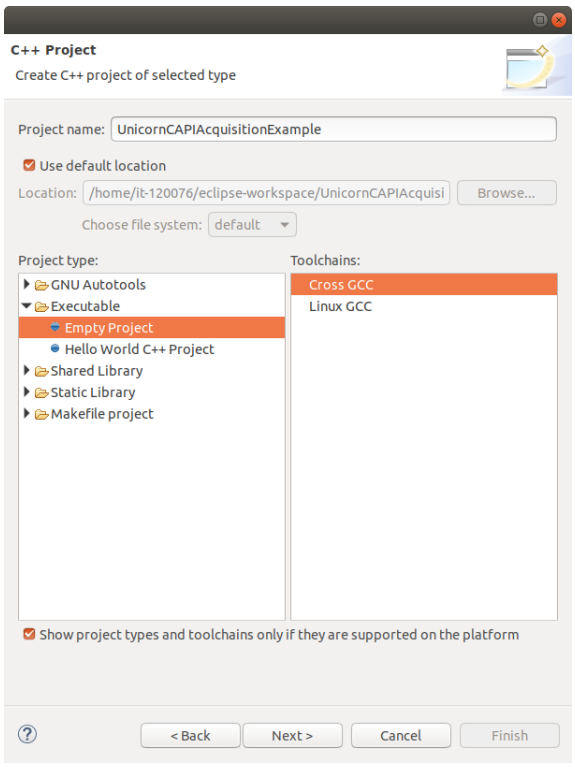
1.3. SETTING UP A PROJECT USING ECLIPSE

1. Open Eclipse.
2. Create a new C++ Console application (File → New → Project → C/C++ Project → C++ Managed Build).

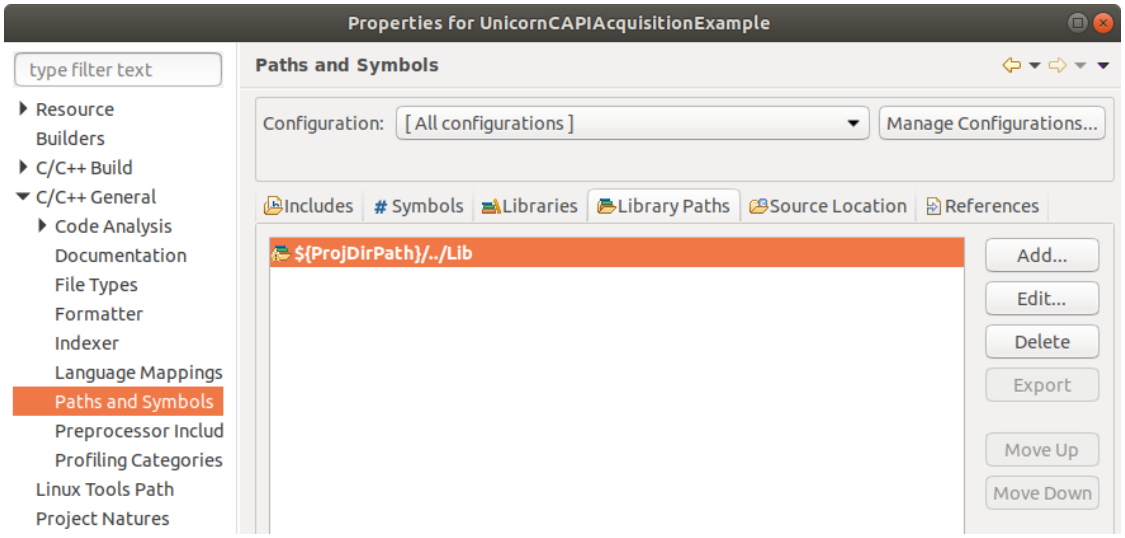




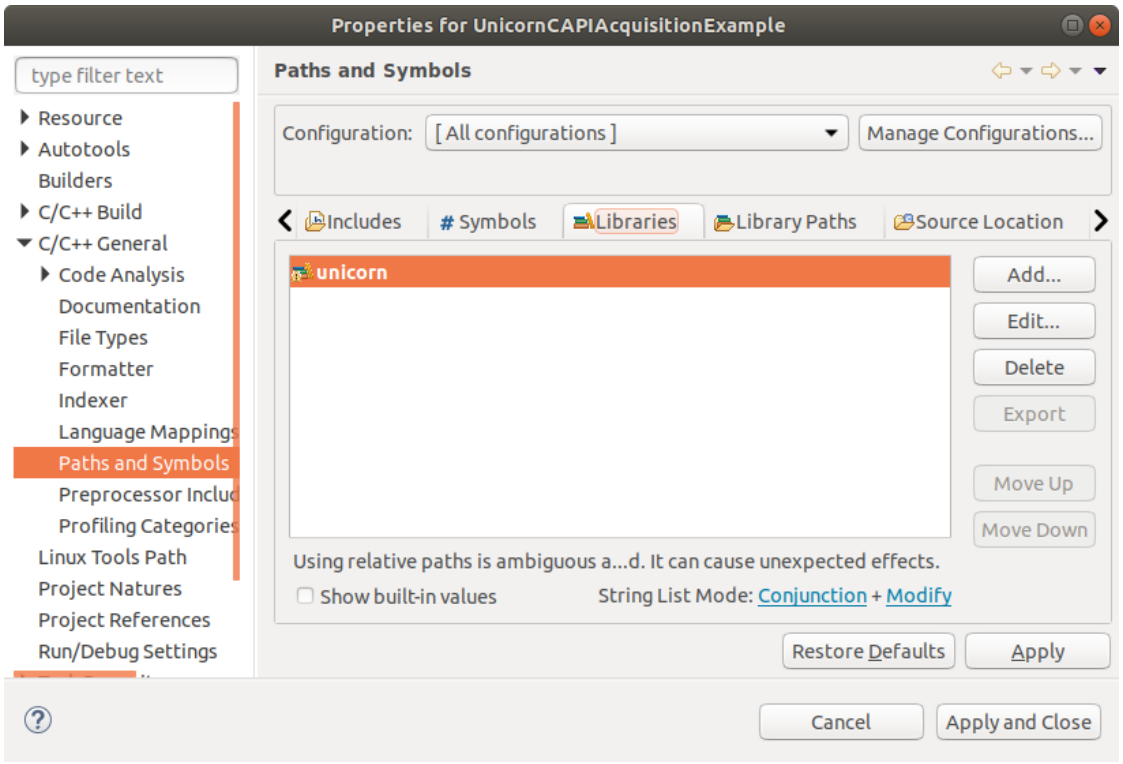
3. Select "Executable" as "Project type" and create an "Empty Project".



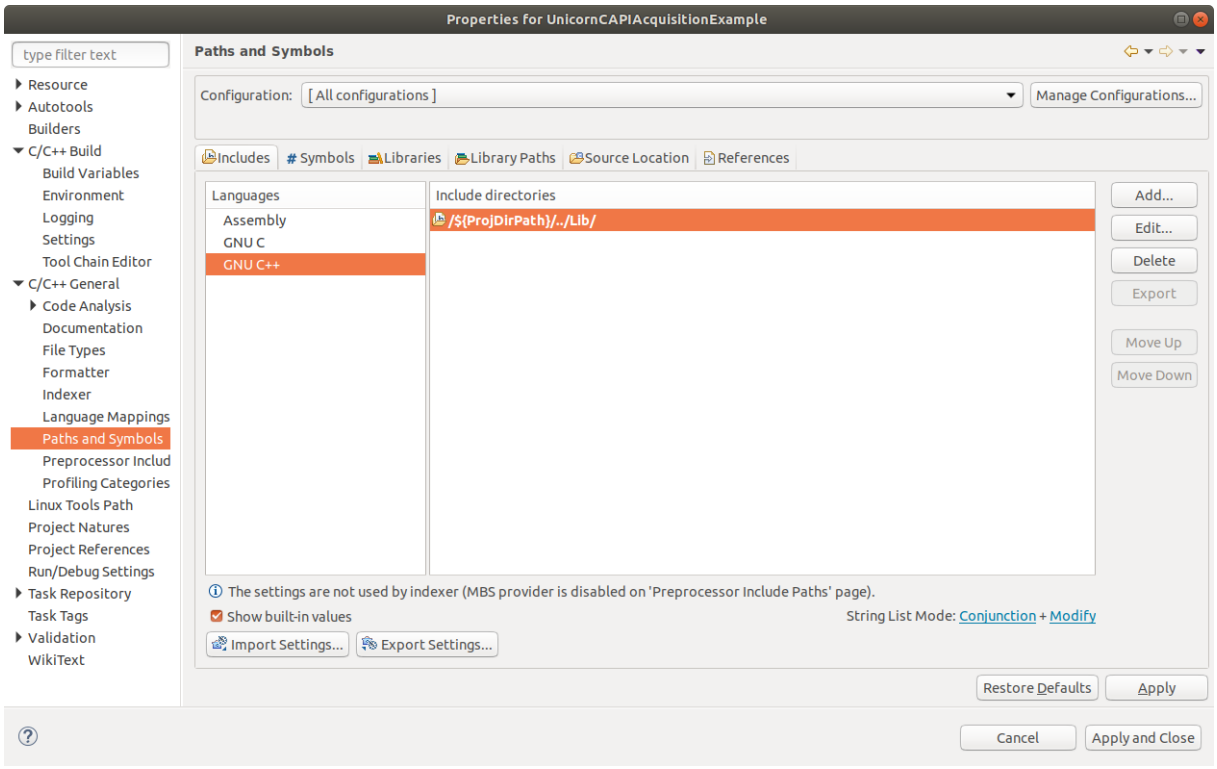
4. Add the "Lib" folder to the search path of the linker. (Properties → C/C++ General → Paths and Symbols → Library Paths)



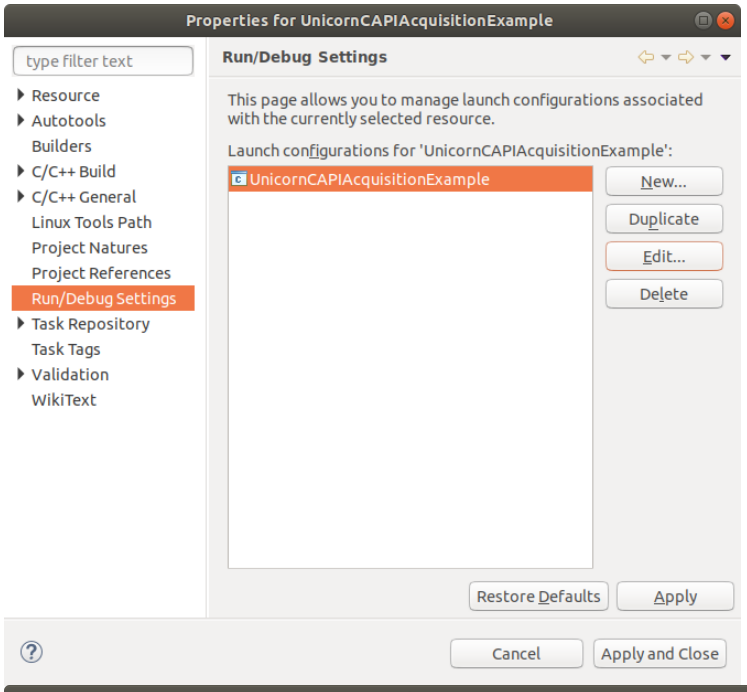
5. Add the "unicorn" library (Properties → C/C++ General → Paths and Symbols → Libraries).

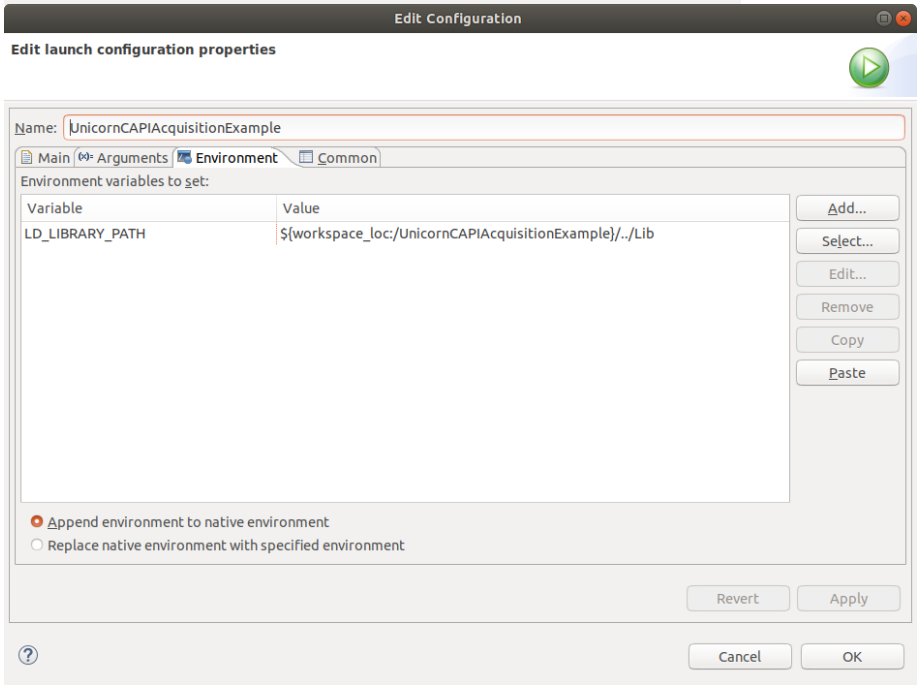


6. Add the "Lib" folder to the "Include" search paths for "GNU C++".

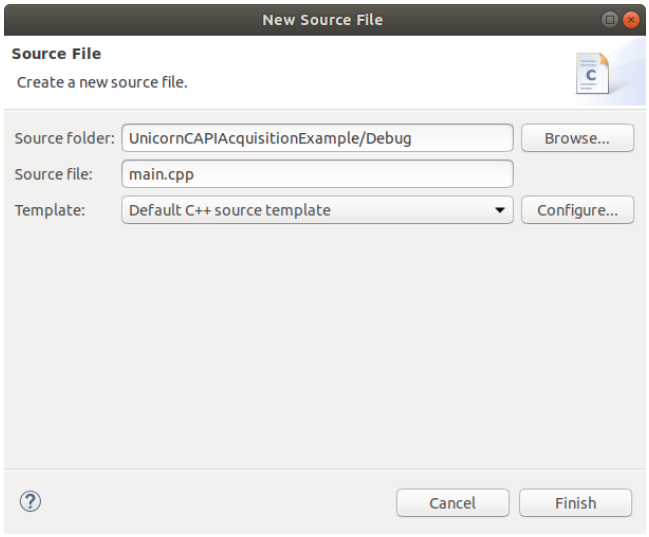


7. Set the "LD_LIBRARY_PATH" environment variable to the Lib folder path to be able to load the unicorn library when debugging or executing your program from eclipse (Properties → Run/Debug Settings → Edit → Environment → Add → "Variable" "LD_LIBRARY_PATH" → "Value" Lib folder path →).


Properties for UnicornCAPIAcquisitionExample
type filter text
Resource
Autotools
Builders
C/C++ Build
C/C++ General
Linux Tools Path
Project Natures
Project References
Run/Debug Settings
Task Repository
Task Tags
Validation
WikiText
Run/Debug Settings
This page allows you to manage launch configurations associated with the currently selected resource.
Launch configurations for 'UnicornCAPIAcquisitionExample':
UnicornCAPIAcquisitionExample
New...
Duplicate
Edit...
Delete
Restore Defaults
Apply
Cancel
Apply and Close


Edit Configuration
Edit launch configuration properties
Name: UnicornCAPIAcquisitionExample
Main
Arguments
Environment
Common
Environment variables to set:
Variable
Value
LD_LIBRARY_PATH
${workspace_loc:/UnicornCAPIAcquisitionExample}/../Lib
Add...
Select...
Edit...
Remove
Copy
Paste
Append environment to native environment
Replace native environment with specified environment
Revert
Apply
Cancel
OK

8. Add a main file to your project



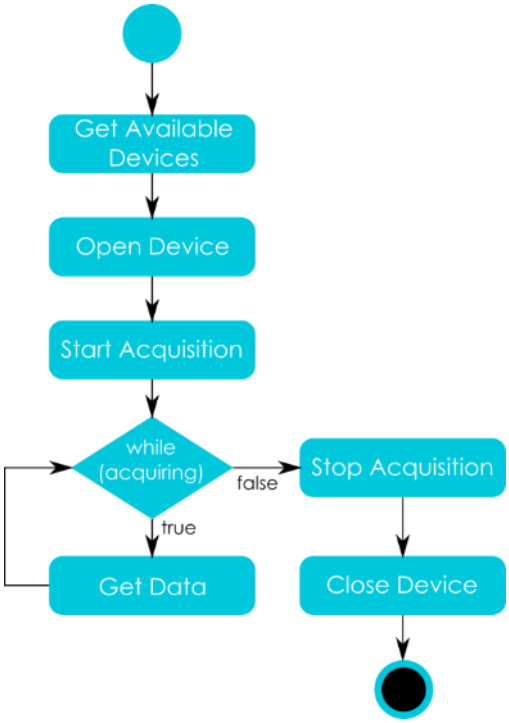
9. You should be able to use the Unicorn Linux C API now

```
// Include unicorn header-file.
#include "unicorn.h"

int main()
{
   // Insert your code here.
   return 0;
}
```

1.4. COMMAND ORDER

To perform a data acquisition using the Unicorn C API, a defined command execution order is required.



1. Before connecting to a Unicorn Brain Interface, it is possible to check the operating environment of Unicorn Brain Interfaces and to discover available Unicorn Brain Interfaces.
2. A connection has to be established to communicate with the Unicorn brain interface. This can be performed by calling Open Device. If a Unicorn Brain Interface handle unequal to Null is received, the connection attempt was executed successfully. After connecting to a Unicorn Brain Interface, it is possible to interact with the Unicorn Brain Interface and call all functions that require a UNICORN_HANDLE. For example, it is possible to read the current configuration of the Unicorn Brain Interface, set a new configuration or start data acquisition.
3. To start data acquisition, Start_Acquisition must be called. After calling Start Acquisition, the Unicorn Brain Interface is set into acquisition mode and is continuously sending data.
4. Therefore, it is required to read the incoming data stream continuously by calling Get Data within an acquisition loop. Other API calls are not allowed while data acquisition is running.
5. To stop data acquisition, Stop Acquisition must be called. The Unicorn Brain Interface will terminate the data stream. The Unicorn Brain Interface is still connected. It is possible to interact with the Unicorn Brain Interface and call all functions that require a UNICORN_HANDLE.
6. To disconnect from a Unicorn Brain Interface, Close_Device must be called. Afterwards, it is not possible to interact with the Unicorn Brain Interface anymore. The Unicorn Brain Interface has to be opened again for interaction.

1.5. UNICORN C API – C REFERENCE

1.5.1. CONSTANTS

UNICORN_ACCELEROMETER_CHANNELS_COUNT

The number of available accelerometer channels.

UNICORN_ACCELEROMETER_CONFIG_INDEX

Index of the first accelerometer UNICORN_AMPLIFIER_CHANNEL in the UNICORN_AMPLIFIER_CONFIGURATION channels array.

UNICORN_BATTERY_CONFIG_INDEX

Index of the battery level UNICORN_AMPLIFIER_CHANNEL in the UNICORN_AMPLIFIER_CONFIGURATION channels array.

UNICORN_COUNTER_CONFIG_INDEX

Index of the counter UNICORN_AMPLIFIER_CHANNEL in the UNICORN_AMPLIFIER_CONFIGURATION channels array.

UNICORN_DEVICE_VERSION_LENGTH_MAX

The maximum length of the Unicorn Brain Interface version.

UNICORN_EEG_CHANNELS_COUNT

The number of available EEG channels.

UNICORN_EEG_CONFIG_INDEX

Index of the first EEG UNICORN_AMPLIFIER_CHANNEL in the UNICORN_AMPLIFIER_CONFIGURATION channels array.

UNICORN_FIRMWARE_VERSION_LENGTH_MAX

The maximum length of the firmware version.

UNICORN_GYROSCOPE_CHANNELS_COUNT

The number of available gyroscope channels.

UNICORN_GYROSCOPE_CONFIG_INDEX

Index of the first gyroscope UNICORN_AMPLIFIER_CHANNEL in the UNICORN_AMPLIFIER_CONFIGURATION channels array.

UNICORN_NUMBER_OF_DIGITAL_OUTPUTS

The number of digital output channels.

UNICORN_SAMPLING_RATE

The sampling rate of the Unicorn Brain Interface.

UNICORN_SERIAL_LENGTH_MAX

The maximum length of the serial number.

UNICORN_STRING_LENGTH_MAX

The maximum string length.

UNICORN_SUPPORTED_DEVICE_VERSION

The Unicorn Brain Interface version which is valid for this API.

UNICORN_TOTAL_CHANNELS_COUNT

The total number of available channels.

UNICORN_VALIDATION_CONFIG_INDEX

Index of the validation indicator UNICORN_AMPLIFIER_CHANNEL in the UNICORN_AMPLIFIER_CONFIGURATION channels array.

1.5.2. ERROR CODES

UNICORN_ERROR_BLUETOOTH_INIT_FAILED

The initialization of the Bluetooth adapter failed.

UNICORN_ERROR_BLUETOOTH_SOCKET_FAILED

The operation could not be performed because the Bluetooth socket failed.

UNICORN_ERROR_BUFFER_OVERFLOW

The acquisition buffer is full.

UNICORN_ERROR_BUFFER_UNDERFLOW

The acquisition buffer is empty.

UNICORN_ERROR_CONNECTION_PROBLEM

The operation could not complete because of connection problems.

UNICORN_ERROR_GENERAL_ERROR

An unspecified error occurred.

UNICORN_ERROR_INVALID_CONFIGURATION

The configuration is invalid.

UNICORN_ERROR_INVALID_HANDLE

The specified Unicorn handle is invalid.

UNICORN_ERROR_INVALID_PARAMETER

One of the specified parameters does not contain a valid value.

UNICORN_ERROR_OPEN_DEVICE_FAILED

The Unicorn Brain Interface could not be opened.

UNICORN_ERROR_OPERATION_NOT_ALLOWED

The operation is not allowed during acquisition or non-acquisition.

UNICORN_ERROR_SUCCESS

The operation completed successfully. No error occurred.

UNICORN_ERROR_UNSUPPORTED_DEVICE

The Unicorn Brain Interface is not supported with this API (UNICORN_SUPPORTED_DEVICE_VERSION).

1.5.3. TYPE DEFINITIONS

BOOL

The Boolean data type, whose values can be TRUE or FALSE.

FALSE

The FALSE value for the BOOL type.

TRUE

The TRUE value for the BOOL type.

NULL

The null pointer.

UNICORN_DEVICE_SERIAL

The type that holds Unicorn Brain Interface serial.

UNICORN_DEVICE_VERSION

The type that holds Unicorn Brain Interface version.

UNICORN_FIRMWARE_VERSION

The type that holds firmware version.

UNICORN_HANDLE

The type that holds the Unicorn Brain Interface handle associated with a device.

1.5.4. STRUCTURES

UNICORN_AMPLIFIER_CHANNEL

The type containing information about a single channel of the Unicorn Brain Interface.

PUBLIC ATTRIBUTES

- char name[32]
  The channel's name.
- char unit[32]
  The channel's unit.
- float range[2]
  The channel's input range as float array. First entry is min value; second is max value.
- BOOL enabled
  The channel's enabled flag. TRUE to enable the channel; FALSE to disable the channel.

UNICORN_AMPLIFIER_CONFIGURATION

The type holding an Unicorn Brain Interface configuration.

PUBLIC ATTRIBUTES

- UNICORN_AMPLIFIER_CHANNEL Channels[UNICORN_TOTAL_CHANNELS_COUNT]
  The array holding a configuration for each available UNICORN_AMPLIFIER_CHANNEL.

UNICORN_DEVICE_INFORMATION

Type that holds additional information about the Unicorn Brain Interface.

PUBLIC ATTRIBUTES

- uint16_t numberOfEegChannels
  The number of EEG channels.
- UNICORN_DEVICE_SERIAL serial
  The serial number of the Unicorn Brain Interface.

- UNICORN_FIRMWARE_VERSION firmwareVersion
  The firmware version number.
- UNICORN_DEVICE_VERSION deviceVersion
  The Unicorn Brain Interface version number.
- uint8_t pcbVersion[4]
  The PCB version number.
- uint8_t enclosureVersion[4]
  The enclosure version number.

1.5.5. FUNCTIONS

1.5.5.1. Close Device

int UNICORN_CloseDevice(UNICORN_HANDLE *hDevice)

Closes a Unicorn Brain Interface.

Disconnects from a Unicorn Brain Interface by a given Unicorn handle.

PARAMETERS

hDevice	A pointer to the handle associated with the session.

RETURNS

An error code is returned as an integer if the disconnection attempt fails.

1.5.5.2. Get API Version

float UNICORN_GetApiVersion()

Returns the current API version.

RETURNS

The current API version.

1.5.5.3. Get Available Devices

int UNICORN_GetAvailableDevices(UNICORN_DEVICE_SERIAL *availableDevices, uint32_t *availableDevicesCount, BOOL onlyPaired)

Scans for available Unicorn Brain Interfaces.

Discovers available paired or unpaired Unicorn Brain Interfaces. Estimates the number of available paired or unpaired Unicorn Brain Interfaces and returns information about discovered Unicorn Brain Interfaces.

PARAMETERS

availableDevices	A pointer to the beginning of an array of UNICORN_DEVICE_SERIAL, which receives available Unicorn Brain Interfaces when the method returns. If NULL is passed, the number of available devices is returned only to determine the amount of memory to allocate.
availableDevicesCount	A pointer to a variable that receives the number of available devices.
onlyPaired	A flag determining whether a full device scan should be performed or not. A quick-scan returns the result of the last scan an is faster than a rescan. A rescan lasts about 10 seconds. If TRUE, a rescan is performed. If FALSE, a quick-scan is performed.

RETURNS

An error code is returned as integer if scanning for available devices fails.

1.5.5.4. Get Channel Index

int UNICORN_GetChannelIndex(UNICORN_HANDLE hDevice, const char *name, uint32_t *channelIndex)

Determines the index of the requested channel within an acquired scan.

Uses the currently set UNICORN_AMPLIFIER_CONFIGURATION to get the index of the requested channel within an acquired scan.

The default names are:

- EEG 1|2|3|4|5|6|7|8
- Accelerometer X|Y|Z
- Gyroscope X|Y|Z
- Counter
- Battery Level
- Validation Indicator

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.
name	The name of the requested channel.
channelIndex	A pointer to a variable that receives the zero-based channel index.

RETURNS

An error code is returned as integer if the index could not be determined.

1.5.5.5. Get Configuration

int UNICORN_GetConfiguration(UNICORN_HANDLE hDevice, UNICORN_AMPLIFIER_CONFIGURATION *configuration)

Gets the current Unicorn Brain Interface configuration.

Retrieves the current Unicorn Brain Interface configuration from the device as UNICORN_AMPLIFIER_CONFIGURATION.

PARAMETERS:

hDevice	The UNICORN_HANDLE associated with the session.
configuration	A pointer to a UNICORN_AMPLIFIER_CONFIGURATION that receives the configuration of the Unicorn Brain Interface.

RETURNS

An error code is returned as an integer if the configuration could not be read.

1.5.5.6. Get Data

int UNICORN_GetData(UNICORN_HANDLE hDevice, uint32_t numberOfScans, float *destinationBuffer, uint32_t destinationBufferLength)

Reads a specific number of scans into the specified destination buffer of known length. Checks whether the destination buffer is big enough to hold the requested number of scans.

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.
numberOfScans	The number of scans to read. The number of scans must be greater than zero. A scan consists of one 32-bit floating point number for each currently acquired channel.
destinationBuffer	A pointer to the destination buffer that receives the acquired data. The destination buffer must provide enough memory to hold the requested number of scans multiplied by the number of acquired channels. Call UNICORN_GetNumberOfAcquiredChannels to determine the number of acquired channels. Call UNICORN_GetChannelIndex to determine the index of a channel within a scan. Example: The sample of the battery level channel in the n-th scan is: n*UNICORN_GetNumberOfAcquiredChannels()+UNICORN_GetChannelIndex("Battery Level")
destinationBufferLength	Number of floats fitting into the destination buffer.

RETURNS

An error code is returned as integer if data could not be read.

1.5.5.7. Get Device Information

int UNICORN_GetDeviceInformation(UNICORN_HANDLE hDevice, UNICORN_DEVICE_INFORMATION *deviceInformation)

Reads the device information by a given UNICORN_HANDLE.

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.
deviceInformation	A pointer to a UNICORN_DEVICE_INFORMATION that receives information about the device.

RETURNS

An error code is returned as an integer if the device information could not be read.

1.5.5.8. Get Digital Outputs

int UNICORN_GetDigitalOutputs(UNICORN_HANDLE hDevice, uint8_t *digitalOutputs)

Reads the current state of the digital outputs.

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.
digitalOutputs	A pointer to a variable that receives the states of the digital output channels. Each bit represents one digital output channel. If a bit is set, the corresponding digital output channel's value is set to high. If a bit is cleared, the corresponding digital output channel's value is set to low. Examples (the binary representation of each decimal value is shown in parentheses): 0 ($0000\ 0000_b$) → all digital outputs set to low. 170 ($1010\ 1010_b$) → digital outputs 2,4,6,8 are set to high. 255 ($1111\ 1111_b$) → all digital outputs set to high.

RETURNS

An error code is returned as an integer if the state of the digital output channels could not be read.

1.5.5.9. Get Last Error Text

const char* UNICORN_GetLastErrorText()

Returns the description of the last error occurred.

RETURNS

The description of the last error occurred.

1.5.5.10. Get Number of Acquired Channels

int UNICORN_GetNumberOfAcquiredChannels(UNICORN_HANDLE hDevice, uint32_t *numberOfAcquiredChannels)

Determines the number of acquired channels.

Uses the currently set UNICORN_AMPLIFIER_CONFIGURATION to get the number of acquired channels.

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.
numberOfAcquiredChannels	A pointer to a variable that receives the number of acquired channels.

RETURNS

An error code is returned as an integer if the number of acquired channels could not be determined.

1.5.5.11. Open Device

int UNICORN_OpenDevice(const char *serial, UNICORN_HANDLE *hDevice)

Connects to a certain Unicorn device and assigns a Unicorn handle if the connection attempt succeeded.

PARAMETERS

serial	The serial number of the device to connect to.
hDevice	A pointer to a UNICORN_HANDLE that receives the handle associated with the current session if the device could be opened successfully.

RETURNS

An error code is returned as an integer if the device could not be opened.

1.5.5.12. Set Configuration

int UNICORN_SetConfiguration(UNICORN_HANDLE hDevice, UNICORN_AMPLIFIER_CONFIGURATION *configuration)

Sets an UNICORN_AMPLIFIER_CONFIGURATION.

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.
configuration	A pointer to the UNICORN_AMPLIFIER_CONFIGURATION to set.

RETURNS

An error code is returned as an integer if configuration is invalid or could not be set.

1.5.5.13. Set Digital Outputs

int UNICORN_SetDigitalOutputs(UNICORN_HANDLE hDevice, uint8_t digitalOutputs)

Sets the digital outputs to high or low.

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.
digitalOutputs	The state of the digital output channels to set in bits. Each bit represents one digital output channel. Set a bit to set the corresponding digital output channel's value to high. Clear a bit to set the corresponding digital output channel's value to low. Examples (the binary representation of each decimal value is shown in parentheses): 0 ($0000\ 0000_b$) → all digital outputs set to low.

	170 ($1010\ 1010_b$) → digital outputs 2,4,6,8 are set to high. 255 ($1111\ 1111_b$) → all digital outputs set to high.

RETURNS

An error code is returned as an integer if the state of the digital output channels could not be set.

1.5.5.14. Start Acquisition

int UNICORN_StartAcquisition(UNICORN_HANDLE hDevice, BOOL testSignalEnabled)

Starts data acquisition in test signal or measurement mode.

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.
testSignalEnabled	Enables or disables the test signal mode. TRUE to start the data acquisition in test signal mode; FALSE to start the data acquisition in measurement mode.

RETURNS

An error code is returned as an integer if data acquisition could not be started.

1.5.5.15. Stop Acquisition

int UNICORN_StopAcquisition(UNICORN_HANDLE hDevice)

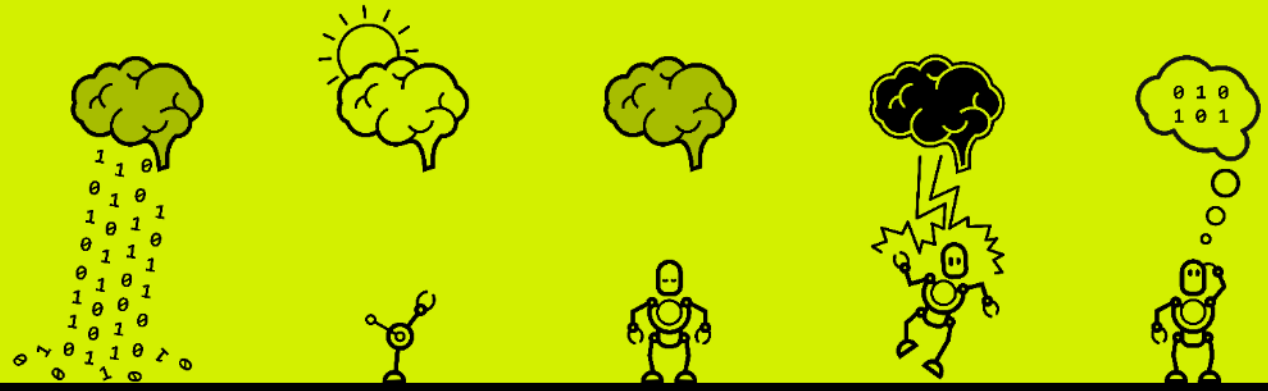
Stops a currently running data acquisition session.

PARAMETERS

hDevice	The UNICORN_HANDLE associated with the session.

RETURNS

An error code is returned as an integer if the acquisition could not be terminated.



BR41N.IO

THE BRAIN-COMPUTER INTERFACE
DESIGNERS HACKATHON

WWW.BR41N.IO

unicorn

THE BRAIN INTERFACE