



Ardulink is a Java Open Source project. Please visit [www.ardulink.org](http://www.ardulink.org) for more information.

Ardulink depends on some Java libraries and native code.

This is a detailed list:

## Serial Connection

Serial Connection is the main connection method to Arduino boards. Ardulink SerialConnection uses serial RXTX library.

RXTX library site is <http://rxtx.qbang.org>

RXTX native code for windows systems (rxtxSerial.dll) is already in Ardulink distribution package. Run set32bitWindowsRXTX.bat or set64bitWindowsRXTX.bat regarding your windows system.

You can install RXTX for other systems reading instructions on RXTX site.

For linux systems you could use this command:

```
sudo apt-get install librxxtx-java
```

For more info about linux systems read this article:

<http://www.ardulink.org/how-to-install-arduino-pc-link-on-raspberry-pi/>

## Bluetooth Connection

Bluetooth Connection is intended for Arduino boards equipped with a bluetooth extension. This connection is developed and tested with an HC-06 board attached to an Arduino UNO. It uses bluecove 2.1.0 java library. In the Ardulink distribution you will find the bluecove jar but not the bluecove GPL jar intended for linux systems. If you need for it, you can download it from bluecove site.

Bluecove site is <http://bluecove.org>

## USB Connection

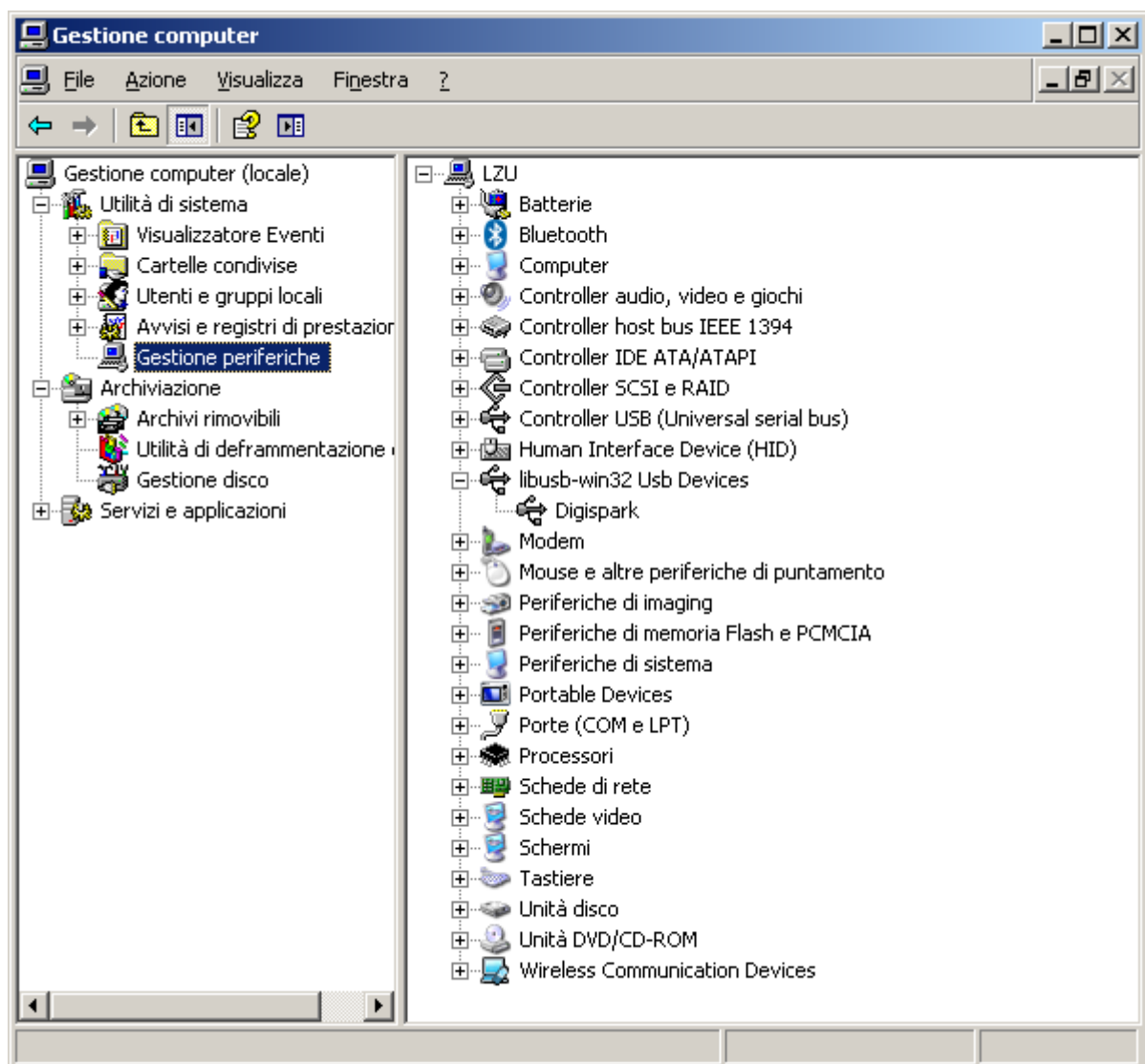
USB connection is intended for Arduino boards that don't have serial connection feature on USB. Actually Ardulink uses this connection for Digispark board ([www.digistump.com](http://www.digistump.com)) and PicoDuino board

([www.tindie.com/products/bobricius/picoduino](http://www.tindie.com/products/bobricius/picoduino)).

Ardulink USB Connection uses Java libusb / libusb-win32 wrapper (<http://libusbjava.sourceforge.net/wp/>). This java library is a wrapper for libusb (0.1) (<http://www.libusb.org/>) and libusb-win32 (<http://libusb-win32.sourceforge.net/>) USB library.

Please check each library's site to install instructions.  
For windows systems you should:

- extract libusb-win32-bin-1.2.6.0.zip
- run inf-wizard.exe
- select Arduino board and finish the wizard
- install USB device using .inf file created



*Picture 1: Device Digispark created*

## Compatibility list

Ardulink is generic and should work with almost any Arduino boards and Arduino clone boards. However, I have tested Ardulink only with some versions of Arduino or Arduino clones.

Actually I tested, in chronological order, Ardulink with:

Board	Site	Ardulink Connection class	Ardulink GUI panel
Arduino UNO	<a href="http://www.arduino.cc">www.arduino.cc</a>	SerialConnection	SerialConnectionPanel
Digispark	<a href="http://digistump.com">digistump.com</a>	DigisparkUSBConnection	DigisparkConnectionPanel (SimpleProtocol)
Arduino Micro	<a href="http://www.arduino.cc">www.arduino.cc</a>	SerialConnection	SerialConnectionPanel
Zigduino r2	<a href="http://logos-electro.com">logos-electro.com</a>	SerialConnection	SerialConnectionPanel
PicoDuino	<a href="#">On tindie</a>	DigisparkUSBConnection	DigisparkConnectionPanel (SimpleProtocol)
ChipKIT Uno32	<a href="http://chipkit.net">chipkit.net</a>	SerialConnection	SerialConnectionPanel
Crowduino With ATmega 328 V1.1	<a href="#">Elecrow</a>	SerialConnection	SerialConnectionPanel
Crowduino Uno-SD V1.4	<a href="#">Elecrow</a>	SerialConnection	SerialConnectionPanel

## Ardulink Console

Ardulink is a java library and you can use it within your java project. However Ardulink has a Console too. So Ardulink can be used from non programmer user too. In order to run Ardulink Console you have just to run ardulink.bat or ardulink.sh scripts shell.

## Ardulink Network Proxy

As explained here:

<http://www.ardulink.org/how-control-arduino-from-network/>

here:

<http://www.ardulink.org/arduino-yun-alternative/>

and here:

<http://www.ardulink.org/control-and-coordinate-many-arduino-boards/>

Ardulink has a Network Proxy that you can use to control Arduino boards over the net.

## Ardulink Sketches

Ardulink needs some code uploaded in Arduino board. Actually there isn't a specific library for Arduino IDE. In Ardulink distribution you can find several sketches that you can use to work with Ardulink. Of course they need for some modifies in order to accomplish your requirements but they are a good start point.

- **ArdulinkProtocol.ino** works with Arduino UNO boards
- **ArdulinkProtocol4Digispark.ino** works with Digispark and PicoDuino using Ardulink default protocol named: ALProtocol. Note: SWING component DigisparkConnectionPanel uses SimpleBinaryProtocol instead of ALProtocol. I suggest you to use SimpleBinaryProtocol with these boards
- **ArdulinkProtocol4LeonardoAndMicro.ino** this is the most general sketch for Ardulink. It is equal to ArdulinkProtocol.ino but manage input in a different way. It is developed to work with Arduino Micro and Leonardo but you can test it with other Arduino based boards.
- **ArdulinkProtocol4Uno32.pde** this sketch is developed to work with chipKIT Uno32 boards
- **CustomMessagesChipKit.ino** this sketch is developed to work with chipKIT Uno32 boards. It manages custom messages, and of course is just an example. I've used it for this video: <https://www.youtube.com/watch?v=PH2ejKGKaoM> where Ardulink manages an OLED installed into the chipKIT Basic I/O Shield
- **JoystickCustomMessages.ino** this is the sketch I've developed to test the release v0.4.2 Top Gun. You can see a video here: [https://www.youtube.com/watch?v=MErhEvY\\_NA8](https://www.youtube.com/watch?v=MErhEvY_NA8)
- **JustReading4Uno32.pde** an example that sends messages to Ardulink reading a PIN
- **SimpleProtocol.ino** a sketch to manage the SimpleBinaryProtocol
- **SimpleProtocol4Digispark.ino** a sketch to manage the SimpleBinaryProtocol with a Digispark or PicoDuino