Leonardo+

1 DESCRIPTION

The PTSolns *Leonardo+* based on the ATmega32u4, is a popular microcontroller development board with additional unique features. The *Leonardo+* has the same footprint as the Uno and continues to be compatible with a range of shields. The board has two extra female header pins that supply SDA and SCL that are shifted from the onboard logic level of 5 V down to 3.3 V. Along with the onboard QWIIC® connector, this shifted 3.3 V I2C bus makes it convenient to connect many common sensors and modules that operate on 3.3 V logic No need for extra external components, all the logic level shifting between 5 V and 3.3 V on the I2C bus is done automatically onboard.

The *Leonardo+* has a modern USB-C port, and the standard power barrel jack onboard. The board has a new layout of components, such as for example, the grouping of the four LEDs (Power LED, Pin 13 controlled LED, TX and RX communication LEDs). Furthermore, the *Leonardo+* allows for a larger current draw on the 3.3 V pin with a rated maximum of 0.8A. This lets the user manage their project's power budget more easily by allowing a larger current draw.

This version of the *Leonardo+* contains the ATmega32U4 IC, perfect for fun projects such as making custom keyboards and mice for computers, among others. The *Leonardo+* comes with a pre-installed out-of-the-box ready set of tests. This gets simple projects started quickly and allows for testing the proper function of the board. A large active online community supports this microcontroller and its programming using the Arduino IDE software. Many tutorials, projects, schematics, and sketches can be easily found online in blogs, forums and the like. This *Plus* version offers all of the features users come to learn and expect and adds additional ones that make this board stand out.

Table of Contents

2 DOCUMENT REVISION HISTORY

Current document revision is Rev 0.

3 PRODUCT FEATURES

This section highlights notable features of the *Leonardo+.*

3.1 I/O Connectors & Input Power

The *Leonardo+* has three input/output (I/O) connectors as follows:

- USB-C
- Power barrel jack (2.1 mm x 5.5 mm)
- QWIIC®

These connectors are shown in Figure 1. Note that the QWIIC connector adheres to the standardization as outlined by SparkFun Electronics: https://www.sparkfun.com/qwiic

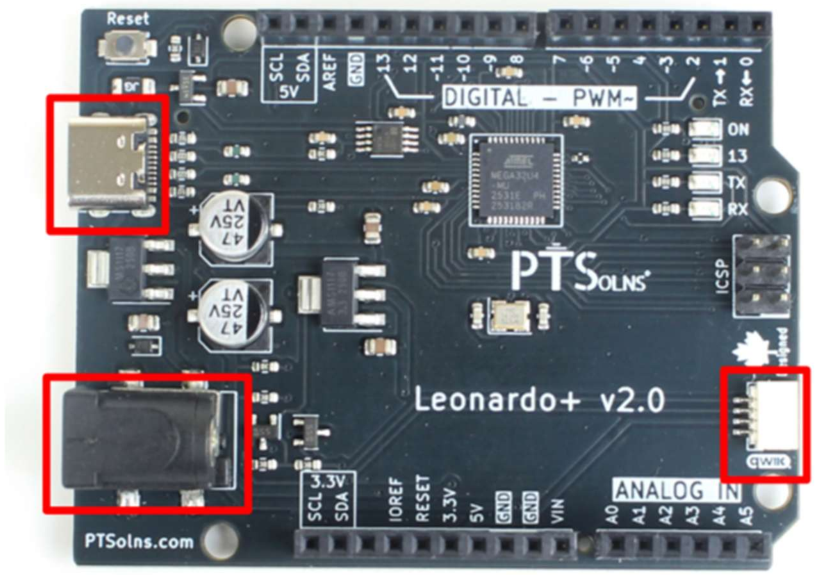


Figure 1: The three I/O connectors on the Leonardo+.

The *Leonardo+* can be powered in multiple ways, with the most common two methods being the USB-C port and the power barrel jack. If the user connects both at the same time, the board automatically selects the voltage from either source depending on the voltage level of the power jack. The user can also power the board using the “VIN” pin on the female header with the same voltage input ratings as for the power barrel jack (i.e. Vin = 7-12 V max).

The board can also be powered directly by supplying a clean and steady 5 V DC to the “5V” pin on the female header. This bypasses the onboard 5 V voltage regulator, and can be done if an external 5 V source is available. **Note that the ATmega328P should not exceed an input voltage of 5.5 V and therefore the user needs to exercise caution when powering the *Leonardo+* from the 5 V pin directly.**

3.1.1 Note on USB-C Cables

From a data transfer perspective USB-C cables can be categorized into two types:

1. Data transfer capable
2. Not data transfer capable

Only the first type of USB-C cable can be used to program not just the *Leonardo+*, but indeed any microcontroller. This type of cable provides power to the board, as well as facilitates data transfer between the computer and the board. **Using this type of cable is essential in programming a microcontroller**.

The second type of USB-C cable does not facilitate the transfer of data but can merely be used to power the board. Therefore, this type of USB-C cable cannot be used to program a microcontroller.

How to tell if a USB-C Cable is Data Transfer Capable?

One can easily and quickly check if a particular USB-C cable is data transfer capable by simply trying to program the *Leonardo+*. If the USB-C cable is not data transfer capable, then upon plugging it into the computer with the other end into the *Leonardo+* no port will appear.

3.2 I2C Bus

The *Leonardo+* offers an I2C bus on the SDA and SCL lines, inherently at the ATmega32U4 5 V logic level. For the ATmega32U4 the 5 V SDA and SCL pins are available on the dedicated SDA and SCL pins on the female header. For convenience these pins are clearly marked as "SCL" and "SDA" in a box with "5 V", as seen in Figure 1. This is to clearly indicate to the user that this part of the I2C bus operates on 5 V logic.

The *Leonardo+* has an onboard Logic Level Shifter (LLS) that shifts the 5 V I2C bus down to 3.3 V. These corresponding I2C pins are available on the female header breakout marked as "SCL" and "SDA" in a box with "3.3 V", as seen in Figure 1. This is to clearly indicate to the user that this part of the I2C bus operates on the 3.3 V logic. The I2C bus at 3.3 V is also made available in the form of a QWIIC® connector. This connector is available on the *Leonardo+* PCB and is marked with the trademark "QWIIC" logo, as seen in Figure 1. Note that the QWIIC connector adheres to the standardization as outlined by SparkFun Electronics:

https://www.sparkfun.com/qwiic

The user can connect multiple peripherals to the *Leonardo+* via I2C simultaneously. It is possible to connect to the 5 V I2C bus, the shifted 3.3 V I2C bus, as well as the QWIIC connector all at once, as long as the following requirements are met:

- The total number of I2C connected peripherals does not exceed 126 units.
- The power budget of the project does not exceed any current rating (see Section 5).
- Each device address is unique (alternatively an I2C multiplexer can be used to connect peripherals with the same address on the same I2C bus).
- There is not excessive capacitance introduced by long connecting wires. The rise-time of the I2C signals, as determined by the corresponding time constant, must be within acceptable range (see discussion below).

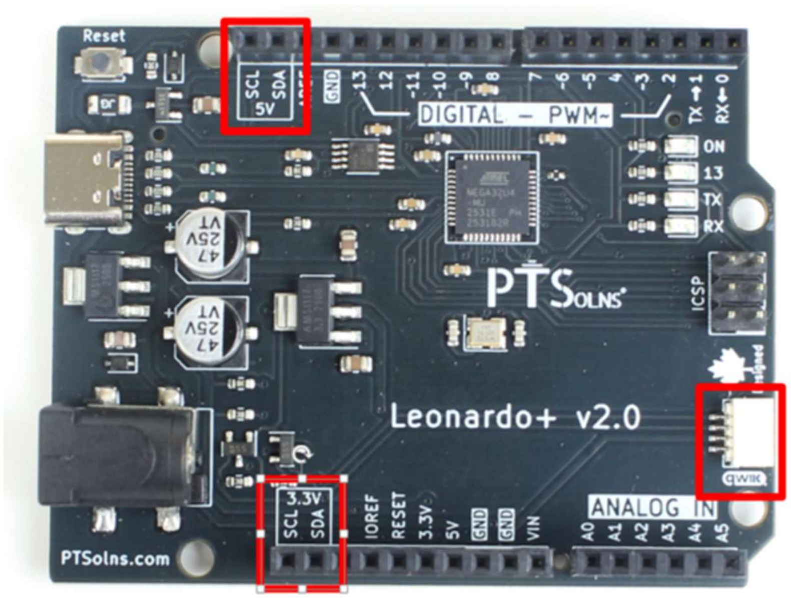


Figure 1: I2C pins at 5 V and 3.3 V, as well as the QWIIC connector, on the Leonardo+.

The default connection for the 3.3 V SDA and SCL pins on the *Leonardo+* is such that it is enabled. This means the *Leonardo+* out-of-the-box will have a functioning 3.3 V I2C bus available on the corresponding female header pins and the onboard QWIIC® connector, as seen in Figure 1. However, it is possible to disconnect these pins by cutting the jumper pads labelled "3.3V SDA" and "3.3V SCL" as shown in Figure 2. When these pins are cut, the 3.3 V SDA and SCL pins and the QWIIC® connector are left floating.

Furthermore, there are additional pull-up resistors on the 5 V I2C bus, each with a value of 10 kΩ. These pull-ups can also be disconnected by cutting the jumper pads labelled "10k Pull-up 5V SCL" and "10k Pull-up 5V SDA" as shown in Figure 2.

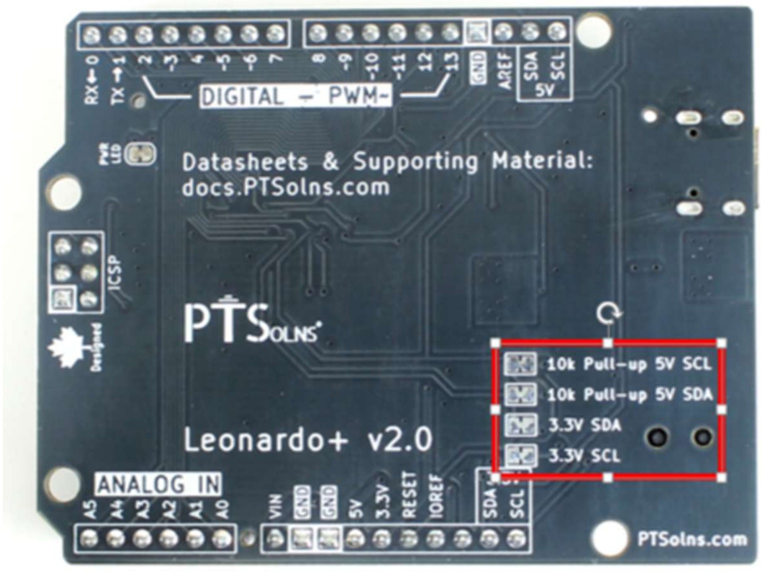


Figure 2: Jumper pads on the back of the Leonardo+ (all closed by default).

The user is made aware that if too many I2C-connected peripherals are used at once, or if the connecting wires are long (around 1.5 m or longer), the I2C communication may start to break down. This is not a deficiency of the *Leonardo+* but rather a characteristic of I2C communication itself. When the SDA signal rise-time starts to take too long and becomes sluggish (because the time constant is too large, which is a function of resistance and capacitance on the I2C bus), data may become corrupt. Often this will result in the microcontroller freezing, and a restart may be required. This scenario is easily diagnosed with an oscilloscope.

The user can try the following simple changes to improve the I2C bus quality:

- Reduce the length of the communication wires on the SDA and SCL lines.
- Add or decrease pull-up resistors on the SDA and SCL lines. Note that many peripherals have their own pull-up resistors onboard, and when connecting many such modules to the I2C bus, the total resistance effectively acts in parallel, thus decreasing the overall resistance. The user may need to reduce the number of pull-ups or disable them entirely. Alternatively, there may not be enough pull-up resistance, in which case a larger resistance may need to be added.
- Reduce the clock speed of the SCL line. The slower the communication speed, the less a sluggish or slow rise-time of the SDA line impacts communication. Typically, the default SCL clock speed is 100 kHz. Although there is no theoretical lower limit for I2C, some peripherals may have issues operating below 10 kHz.

3.3 ICSP Breakout

The *Leonardo+* has an In-Circuit Serial Programming (ICSP) breakout section via a 2X3 Pin male header, as shown in Figure 3. The pin breakout of the ICSP is shown in the schematic in Figure 4, with the following definitions:

- CIPO: Controller In, Peripheral Our (formerly called MISO)
- SCK: Serial Clock
- COPI: Controller Out, Peripheral In (formerly called MOSI)
- Reset: Reset

For orientation of the pins, the ground “GND” pin is outlined with a white printing, best seen from the back of the board.

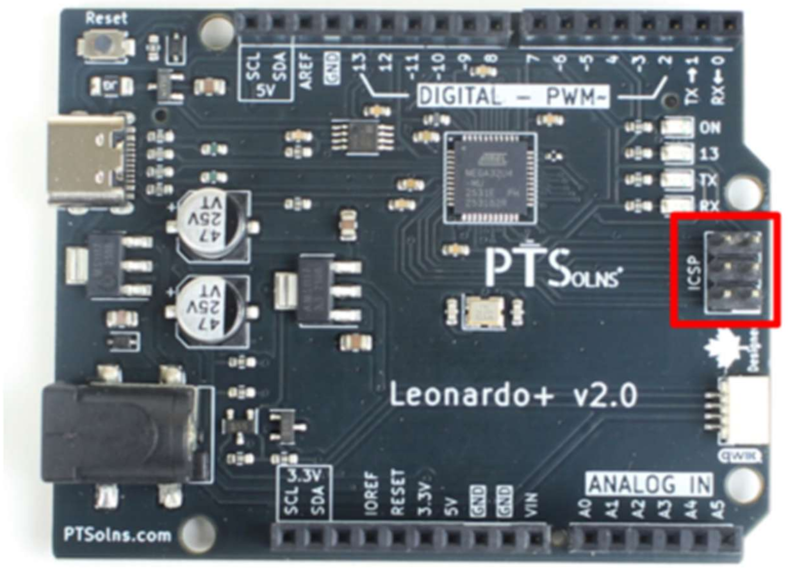


Figure 3: Breakout pins of the Leonardo+.

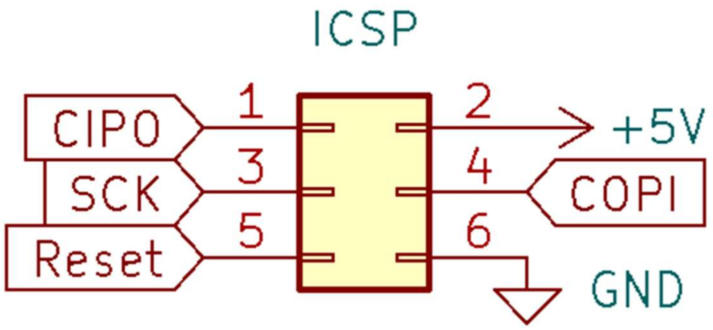


Figure 4: ICSP breakout pins.

3.4 Compatibility with Shields (Stackability)

The *Leonardo+* is fully stackable and thus compatible with a range of common Uno form factor shields. Therefore, all of the regular breakout pins on both the female headers as well as the male header are available, as shown in Figure 4. The *Leonardo+* includes additional pins on the female header labelled "SCL" and "SDA" in a box marked "3.3 V" (see Section 3.2). For most shields, these additional pins do not interfere with a shield stacked on top.

3.5 Component Arrangement & Pinout Diagram

The *Leonardo+* has a unique component arrangement. As an example, the four onboard LEDs (Power LED, Pin 13 controlled LED, TX and RX communication LEDs). Some notable component arrangements, including the LEDs, Reset button and the ATmega32U4, are shown in Figure 5.

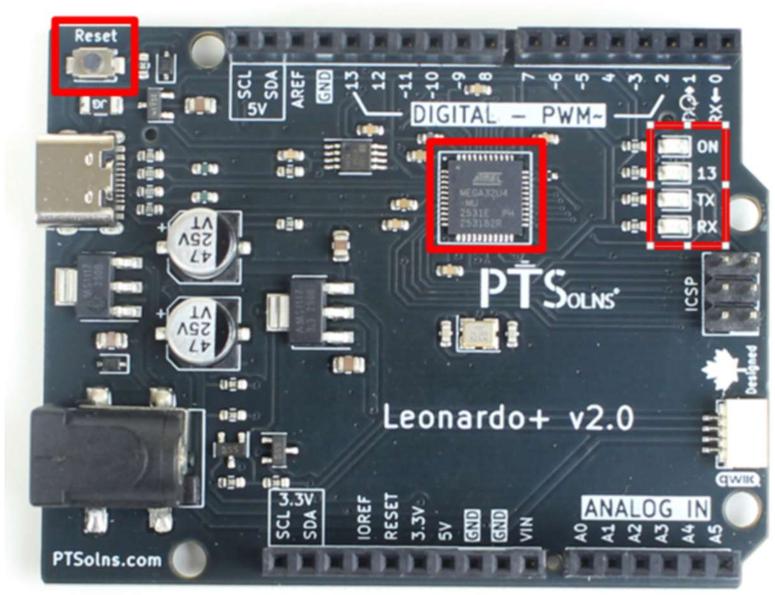


Figure 5: Notable component arrangements of the Leonardo+.

The *Leonardo+* has multiple digital Input/Output (I/O) pins, referred to as "DIO", or just have the prefix "D", of which six are PWM capable. There are multiple analog input pins, and various power related pins. All available pins are shown in the pinout diagram in Figure 6.

Note the following:

- A tilde next to a DIO indicates that pin is PWM capable.
- D13 also controls the onboard LED labelled "13" as shown.
- The Reset pin is connected to the push button as well as the pinout on the ICSP header.

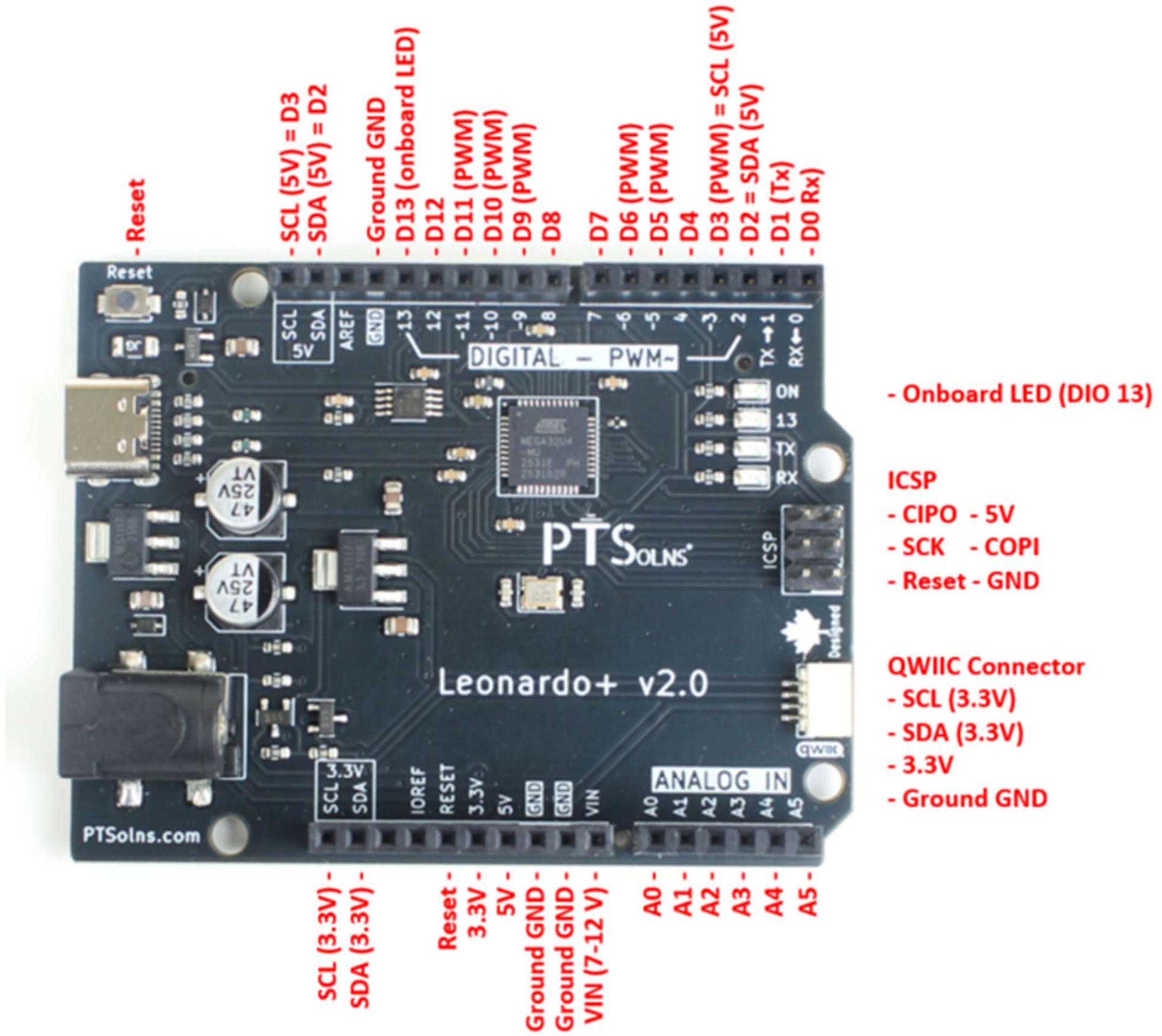


Figure 6: Pinout diagram for the Leonardo+.

3.6 Mark of Authenticity

Authentic PTSolns PCBs have a black solder mask color and are marked with the “PTSolns” logo in white silkscreen printing. The “Canadian Designed” symbol, consisting of the Canadian Maple Leaf with the word “Designed” underneath, can also be found on the PCB in white silkscreen printing. The “PTSolns” trademark and the “Canadian Designed” symbols are shown in Figure 7.

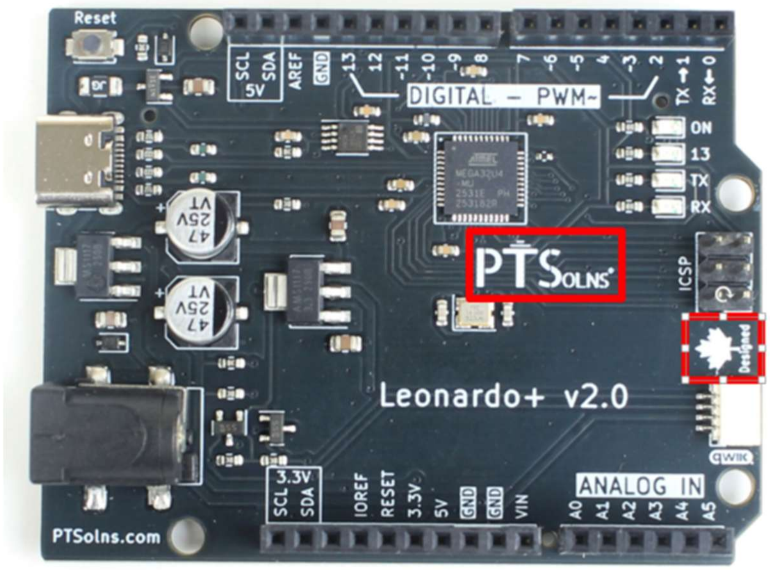


Figure 7: The "Canadian Designed" symbol and the PTSolns trademark found on authentic PTSolns PCBs.

4 PHYSICAL PROPERTIES

The physical properties of the *Leonardo+* are outlined in Table 1.

Table 1: Physical Properties.

	Quantity	**Value**	**Reference**
PCB	Length	68.6 mm	Figure 8
	Width	53.3 mm	Figure 8
	Thickness	1.6 mm	--
	Weight (with headers)	21 g	--
	Color	Black	--
	Silkscreen	White	--
Material	Lead free HASL-RoHS surface finish		--
	FR-4 base		--
Mounting Holes	4x each with 1.651 mm diameter		Figure 9

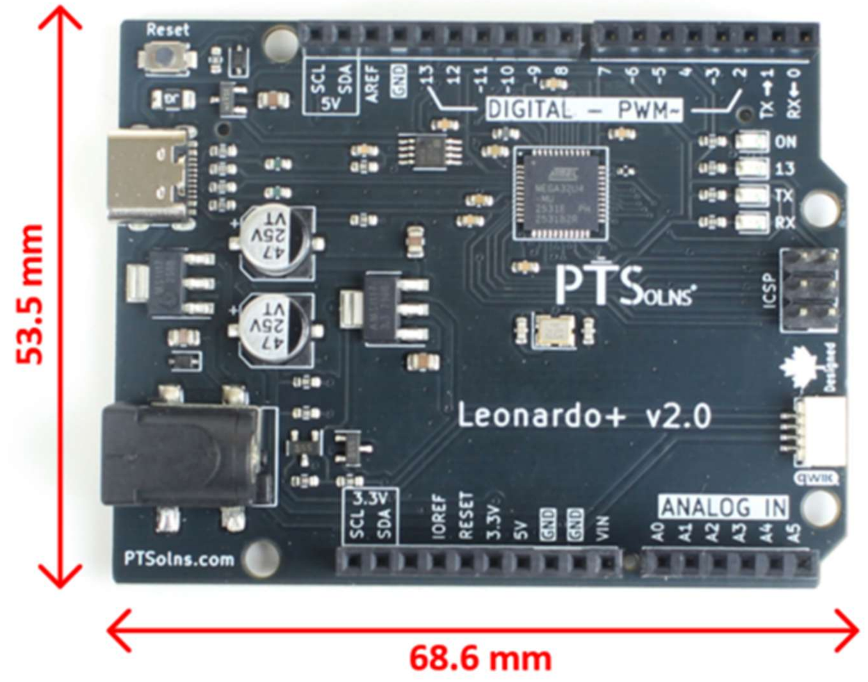


Figure 8: Dimensions of the Leonardo+.

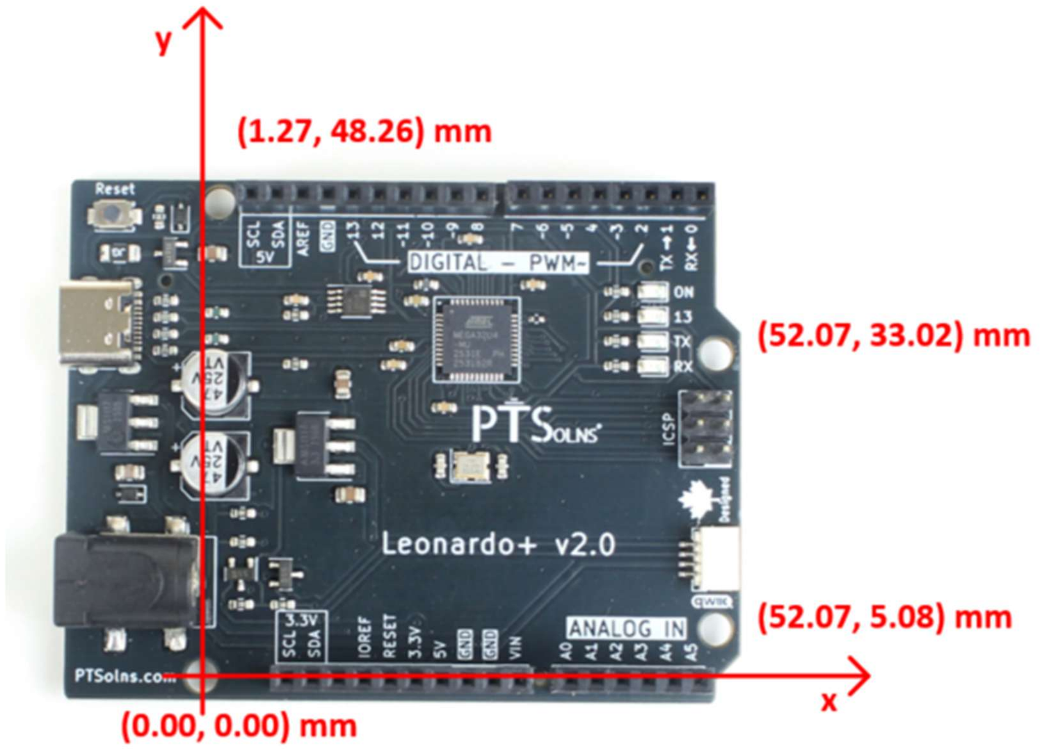


Figure 9: Positions of mounting holes.

5 ELECTRICAL PROPERTIES

The electrical ratings for the *Leonardo+* are outlined in Table 2.

Table 2: Electrical ratings for the *Leoardo+*.

Type	**Rating**
Input voltage on USB-C	5 V
Input voltage on VIN pin	7-12 V
Operating current draw on any single GPIO	20 mA
Absolute max current draw on any single GPIO	40 mA **(Do not operate at this level for extended periods)**
Max combined current draw on all GPIO	200 mA **(IF OPERATING in "Stable" CONDITIONS. See Section 5.1 for important details)**
Max current draw on 3.3 V power pin	160 mA **(IF OPERATING in "Stable" CONDITIONS. See Section 5.1 for important details)**
Max current draw on 5 V power pin	800 mA **(IF OPERATING in "Stable" CONDITIONS. See Section 5.1 for important details)**
Max combined current draw on all GPIO and power pins (Total External Current Draw (TECD))	800 mA **(IF OPERATING in "Stable" CONDITIONS. See Section 5.1 for important details)**

The current ratings for the two power pins, the 3.3 V and 5 V pins, are discussed in more detail in Section 5.1. The voltage ratings are discussed in Section 0.

5.1 Current Rating

The onboard components on the Leonardo+ consume approximately 20 to 40 mA, depending on the LEDs and the ATmega32U4 microcontroller demand. External current draws can be made by employing any of the General-Purpose Input/Output (GPIO) pins, such as the digital or analog pins, as well as the 3.3 V and 5 V power pins. The 3.3 V and 5 V power pins can be used to power external sensors and modules. The combined current draw of any GPIO and power pins used is the Total External Current Draw (TECD).

The TECD is delivered by an onboard 5 V voltage regulator. Depending on the input voltage (Vin) supplied to the regulator, as well as the TECD, the *Leonardo+* may operate in the "Stable" region or the "Unstable" region. This is shown in Table 3. In this context, "Stable" is defined such that the 5 V line remains constant, with a small ripple, within acceptable tolerances. "Unstable" is defined such that the 5 V line starts to drop below unacceptable tolerances, collapses, or rises above 5 V plus an acceptable tolerance. The user should only operate the Leonardo+ in the "Stable" region. In extreme unstable regions (e.g. Vin = 12 V, TECD = 800 mA), the voltage regulator may allow the input voltage through to the 5 V line. This can cause damage to components downstream, and the user must avoid such extreme conditions at all times.

As an example, at an input voltage of Vin = 7 V, the TECD can reach the maximum of 800 mA. With increasing input voltages, the TECD in which the Leonardo+ operates in the "Stable" regions starts to reduce.

Therefore, the user should take care that all current draws on external pins (GPIO, 3.3 V and 5 V power pins) remain in the "Stable" region for a given input voltage Vin, as outlined in Table 3.

Table 3: TECD Operating Conditions.

Total External Current Draw (TECD)
(Not including current draw of onboard components)

Vin	0.0 A	0.1 A	0.2 A	0.3 A	0.4 A	0.5 A	0.6 A	0.7 A	0.8 A
7.0 V	Stable	Stable	Stable	Stable	Stable	Stable	Stable	Stable	Stable
7.5 V	Stable	Stable	Stable	Stable	Stable	Stable	Stable	Unstable	DNO
8.0 V	Stable	Stable	Stable	Stable	Stable	Unstable	DNO	DNO	DNO
8.5 V	Stable	Stable	Stable	Stable	Unstable	DNO	DNO	DNO	DNO
9.0 V	Stable	Stable	Stable	Stable	Unstable	DNO	DNO	DNO	DNO
9.5 V	Stable	Stable	Stable	Unstable	DNO	DNO	DNO	DNO	DNO
10.0 V	Stable	Stable	Stable	Unstable	DNO	DNO	DNO	DNO	DNO
10.5 V	Stable	Stable	Stable	Unstable	DNO	DNO	DNO	DNO	DNO
11.0 V	Stable	Stable	Stable	Unstable	DNO	DNO	DNO	DNO	DNO
11.5 V	Stable	Stable	Unstable	DNO	DNO	DNO	DNO	DNO	DNO
12.0 V	Stable	Stable	Unstable	DNO	DNO	DNO	DNO	DNO	DNO

Stable	Voltage on 5 V line remains stable.
Unstable	Voltage on 5 V line collapses and becomes unstable.
DNO	Do No Operate!

NOTE 1: Drawing full allowed current and temperature of 5 V voltage regulator

If the TECD is high the voltage regulator will get hot. This is unavoidable and depends greatly on the input voltage. The component's temperature depends on how much power (in Watts) it has to dissipate. This power is a result of the voltage difference between the Vin and Vout of the voltage regulator, times the current being draw. This can be expressed in the following formula:

$$Power_{dissipate\ on\ volt.reg} = (V_{in} - V_{out}) * I$$

The input voltage (discussed further in Section 5.2) ranges from 7 V to 12 V. The condition that results in the highest amount of power dissipation in the voltage regulator occurs when Vin = 12 V and the TECD is at its maximum of 800 mA. In this case, the power that must be dissipated becomes (12 V − 5 V) × 800 mA = 5.6 W. This is a significant amount of power for a small surface-mount component, and the component will heat up accordingly. The 5 V voltage regulator will automatically shut down when the internal component temperature reaches approximately 145 °C. The regulator will restart automatically once the temperature falls below a safe threshold.

The *Leonardo+* was developed with temperature and current ratings in mind. The PCB traces that carry higher currents have been made intentionally wide to reduce resistive heating, particularly the trace supplying the 5 V power pin. In addition, thermal vias have been placed beneath both the 5 V and 3.3 V voltage regulators. These vias conduct excess heat from beneath the components to the opposite side of the PCB, where increased surface area allows for more effective heat dissipation.

Copyright © 2026 Pylon Technology Solutions Ltd.

5.2 Voltage Rating

The 5 V voltage regulator determines the maximum and minimum acceptable voltage input a user can supply to the *Leonardo+*. The maximum input voltage is specified as 12 V. However, the regulator can tolerate short momentary voltage spikes of up to 14 V caused by an external power source. The user should take care to never exceed the 12 V rating, as doing so may result in damage to onboard components. If the input voltage is too high, the regulator may malfunction, allowing excessive voltage to appear on the 5 V line. This can cause damage to downstream components on the 5 V rail, including the ATmega32U4. Therefore, when supplying the Leonardo+ with a 12 V source—particularly from a battery or an unstable buck or boost converter—the user must ensure that the voltage does not exceed the 12 V rating.

The minimum input voltage is specified as 7 V. At 7 V the *Leonardo+* can be reliably operated. That being said, the input voltage can likely be reduced slightly below this level, depending on how much current is being drawn through the 5 V voltage regulator as well as the fuse settings of the ATmega32U4. The higher the current draw, the larger the regulator's dropout voltage becomes. At a current draw of 800 mA, the regulator exhibits a voltage drop of approximately 1.45 V. An input voltage of 6 V may therefore be sufficient for very low external current draws. The user is encouraged to experiment carefully to determine whether a lower input voltage is acceptable for their specific project setup.

6 PROGRAMMING

This section explains the first-time setup of the *Leonardo+*, including driver setup, USB-C cable power and data requirements, and programming (uploading a sketch onto the board).

6.1 USB-C Cable, Data Transfer Capable

From a data transfer perspective, USB-C cables can be categorized into two types:

1. Data transfer capable
2. Not data transfer capable

Only the first type of USB-C cable can be used to program not just the *Leonardo+*, but indeed any microcontroller. This type of cable provides power to the board, as well as facilitates data transfer between the computer and the board. **Using this type of cable is essential in programming a microcontroller**.

The second type of USB-C cable does not facilitate the transfer of data but can merely be used to power the board. Therefore, this type of USB-C cable cannot be used to program a microcontroller.

How to tell if a USB-C Cable is Data Transfer Capable?

One can easily and quickly check if a particular USB-C cable is data transfer capable by simply trying to program the *Leonardo+*. If the USB-C cable is not data transfer capable, then upon plugging it into the computer with the other end into the *Leonardo+* no port will appear.

To read more about data transfer capable USB-C cables, the reader is referred to our Tinker Thoughts Blog TTB#10, and related video:

https://ptsolns.com/blogs/tinker-thoughts/ttb10-why-your-usb-c-cable-wont-program-your-microcontroller

6.2 *Leonardo+* USB Enumeration Delay

A short delay on the *Leonardo+* is caused by its native USB architecture and is reflected in how the IDE handles board selection port detection sketch upload and reset events. The microcontroller creates the USB device itself so it first enumerates in bootloader mode then disconnects and re enumerates as the runtime USB interface used by the sketch. This brief disconnect and reconnect appears as a delay in the operating system and IDE. The same sequence occurs after programming or pressing reset and the sketch only starts once USB reinitialization is complete. This behavior is normal and expected and indicates correct operation.

As the microcontroller creates the USB device itself, it does not rely on any external IC to do so, such as the common CH340 USB to serial chip.

6.3 *PTSolns IDE* to Program Development Boards

Programming the *Leonardo+*, in fact most common microcontroller development boards, is easy and intuitive with the programming software *PTSolns IDE*. It is free to download and use. Open-source, community driven, and hosted on GitHub. There are **no sign-ups**, **no sign-in**, and **no subscriptions**. The desktop app is available for Windows, Apple, and Linux operating systems (OS). *PTSolns IDE* can be downloaded from the following link:

https://PTSolns.com/IDE

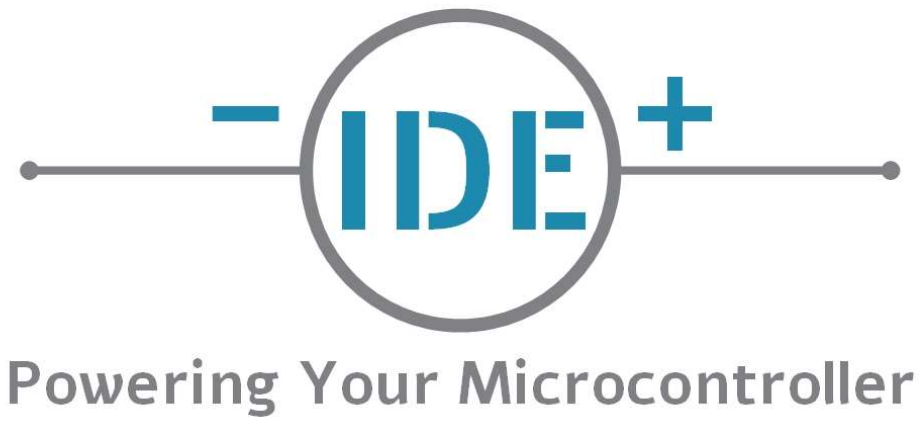


Figure 10: PTSolns IDE - Powering Your Microcontroller

For help with anything related to installation or first time launching the software, please consult this page:

https://PTSolns.com/IDE-Help

6.3.1 Board Selection in *PTSolns IDE*

When first opening *PTSolns IDE* the *Leonardo+* board must be selected. To do so, press the down arrow on the Boards and Ports window, as shown in Figure 11. Upon pressing the little pencil icon, or equivalently the text "Select other board and port…", a new window appears. In the search, type in "leonardo" and the board selection comes up, as shown in Figure 12.

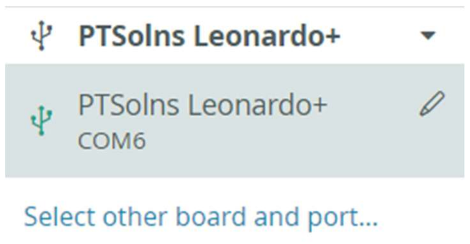
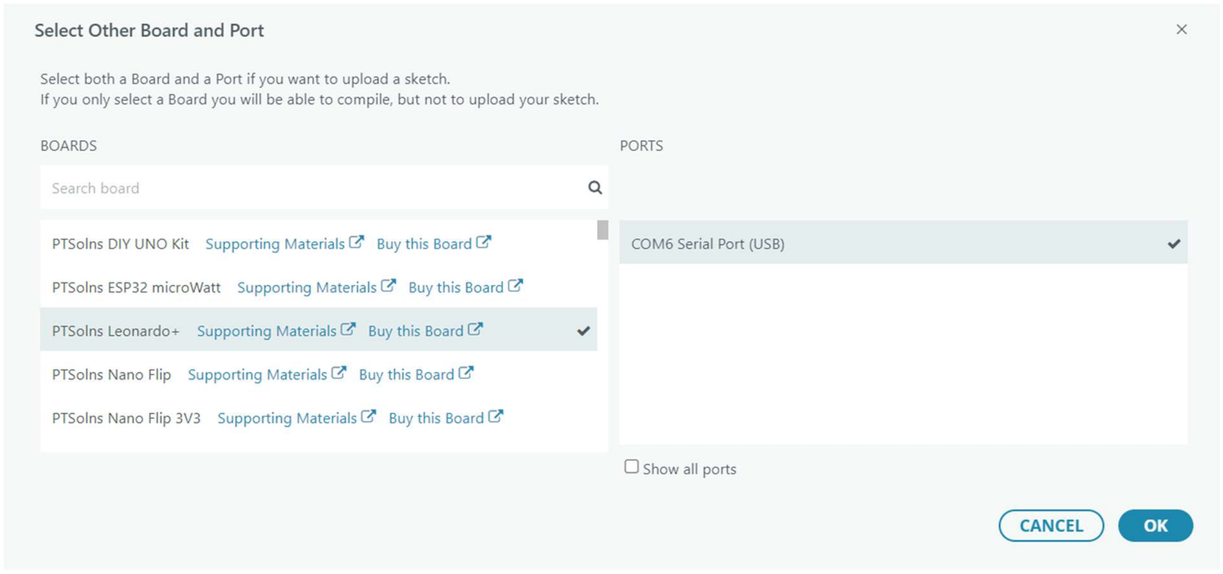


Figure 11: Boards and Ports Selection in PTSolns IDE.


Select Other Board and Port

Select both a Board and a Port if you want to upload a sketch.
If you only select a Board you will be able to compile, but not to upload your sketch.

BOARDS

- PTSolns DIY UNO Kit — Supporting Materials — Buy this Board
- PTSolns ESP32 microWatt — Supporting Materials — Buy this Board
- PTSolns Leonardo+ — Supporting Materials — Buy this Board ✓
- PTSolns Nano Flip — Supporting Materials — Buy this Board
- PTSolns Nano Flip 3V3 — Supporting Materials — Buy this Board

PORTS

- COM6 Serial Port (USB) ✓

☐ Show all ports

Figure 12: Select the Leonardo+ *Board in PTSolns IDE.*

Note that in the image of Figure 12 on the left side is the selection for the BOARDS (*Leonardo+* in this case), and on the right side the selection for the PORTS. The port selection and details relating to it are explained in Section 6.3.2.

6.3.2 Port Selection in *PTSolns IDE*

To be able to see one or more ports in the PORTS selection section in Figure 12 (on the right), the following condition must be true:

1. The *Leonardo+* is plugged into the computer with a USB-C cable that is capable to transfer data. Some USB-C cables only provide power, but is not able to transfer data. For more information on this the user is referred to Section 6.1.

If the above condition is satisfied, the user will be able to see a port selection on the right side of the image in Figure 12. It is possible to see more than one port to choose from, in that case the user can simply unplug and re-plug the board and see which port number momentarily went offline and came back.

6.3.3 Running First Example Sketch *PTSolns IDE*

A common and popular sketch to run on a new development board is the classic “Blink” example. This sketch is useful as the coding is simple, yet the successful upload and running demonstrates critical working parts of the software as well as the board itself. Running Blink demonstrates that:

- *PTSolns IDE* is working properly,
- The CH340 driver is installed and able to recognize the board, and
- The hardware onboard the development board is working (USB to microcontroller, power circuitry)

To select the Blink sketch is simple from within *PTSolns IDE* as it is installed by default. Click on “File” in the top bar menu, then “Examples\01.Basics\Blink”, as shown in Figure 13.

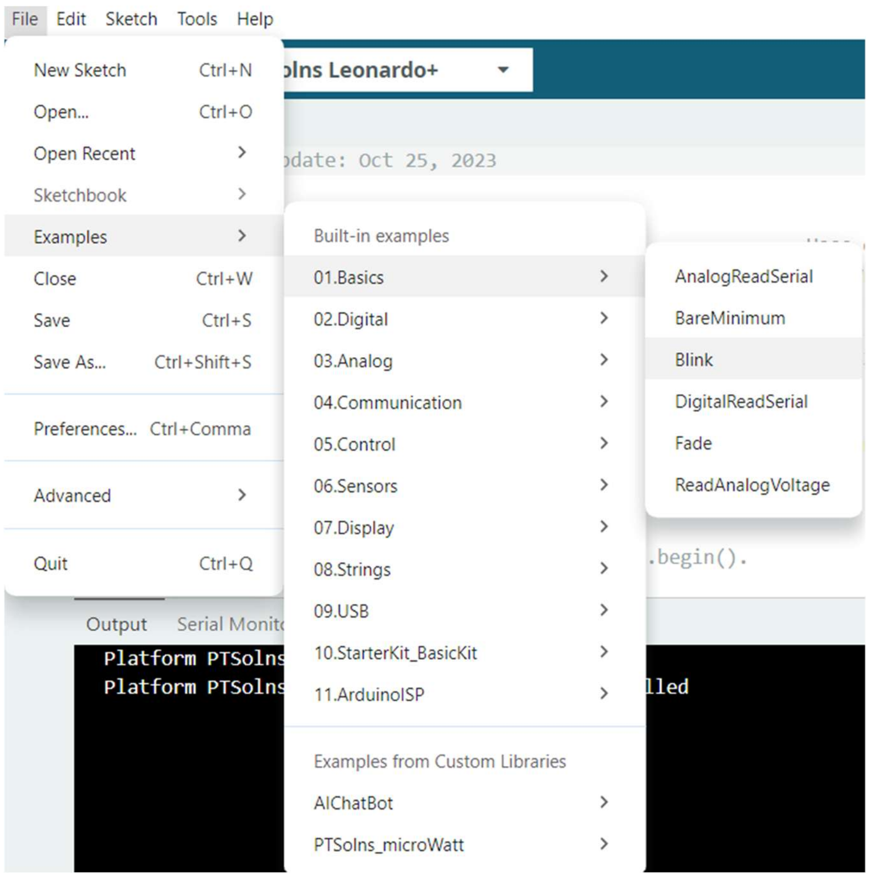


Figure 13: Selecting the "Blink" example from within PTSolns IDE.

6.4 Arduino IDE to Program Development Boards

Arduino IDE is an application to program microcontroller development boards. The application can be downloaded from the following link:

https://www.arduino.cc/en/software

6.4.1 Board Selection in *Arduino IDE*

When first opening Arduino IDE the *Leonardo+* board must be selected. Arduino IDE does not inherently have the exact board selection option, however the board "Arduino Leonardo" works equally well, as shown in Figure 14 and Figure 15.

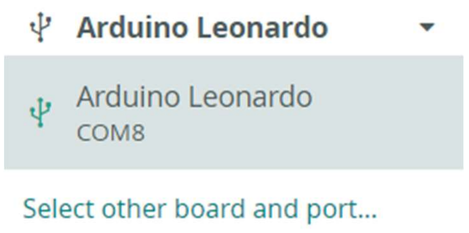


Figure 14: Select other board and port...

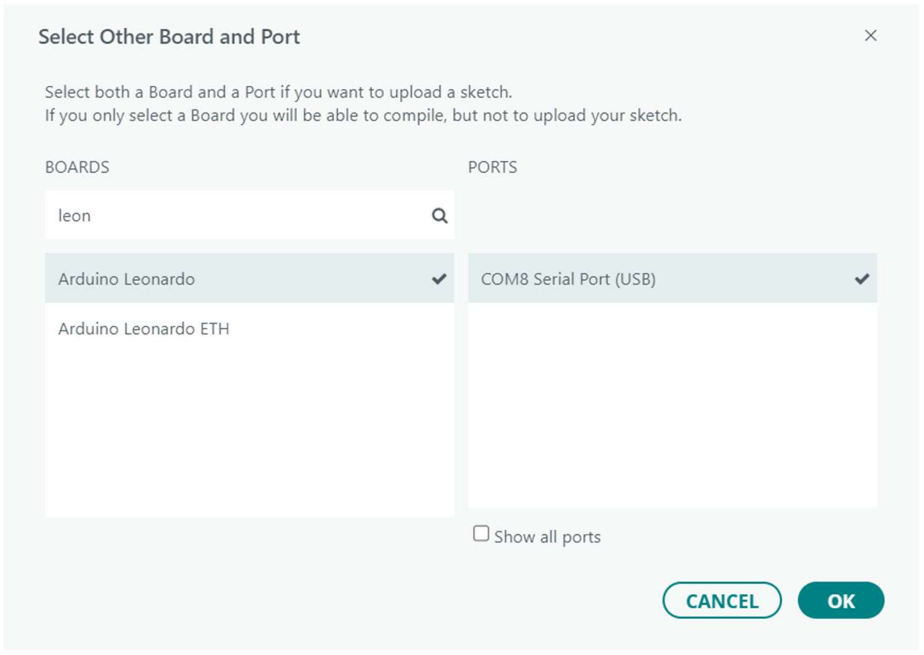

Select Other Board and Port
×
Select both a Board and a Port if you want to upload a sketch.
If you only select a Board you will be able to compile, but not to upload your sketch.
BOARDS
PORTS
leon
Arduino Leonardo ✓
Arduino Leonardo ETH
COM8 Serial Port (USB) ✓
Show all ports
CANCEL
OK

Figure 15: "Arduino Leonardo" board selection for the Leonardo+.

6.4.2 Port Selection in *Arduino IDE*

To be able to see one or more ports in the PORTS selection section in Figure 15 (on the right), the following condition must be true:

1. The *Leonardo+* is plugged into the computer with a USB-C cable that is capable to transfer data. Some USB-C cables only provide power, but is not able to transfer data. For more information on this the user is referred to Section 6.1.

If the above condition is satisfied, the user will be able to see a port selection on the right side of the image in Figure 15. It is possible to see more than one port to choose from, in that case the user can simply unplug and re-plug the board and see which port number momentarily went offline and came back.

6.4.3 Running First Example Sketch *Arduino IDE*

Running the first example is straightforward and similar to *PTSolns IDE*. Therefore, the user is referred to Section 6.3.3.

Copyright © 2026 Pylon Technology Solutions Ltd.

6.5 Out-of-the-Box Ready Examples

The *Leonardo+* comes pre-installed with a sketch that allows the user to perform several tests without any initial software uploading or programming. These pre-programmed tests can be used to check the working order of the *Leonardo+,* or to get started quickly making some simple examples. These out-of-the-box ready tests include:

1) Onboard (and Pin 13) LED blinking in unique pattern.
2) Reset makes onboard LED (and Pin 13) blink in fast pattern of four for one cycle.
3) I2C scanner searches for connected devices (5 V or 3.3 V I2C bus, or QWIIC connector) every 5 seconds and prints the results to Serial monitor on baud rate 9600.
4) Pin 9 is on a fading in and out cycle that can drive an external LED accordingly.
5) Analog read on Pin A0 and displayed to Serial monitor on baud rate 9600.

Each of these tests is explained in further detail below.

6.5.1 Test 1: Onboard LED (Pin 13) Blink Unique Pattern

With the *Leonardo+* powered, the programmable onboard LED (marked “IO13”) blinks in a regular unique pattern. The LED will illuminate for 100 mS and turn OFF for 200 mS. If the reset is pressed (see Test 2) the pattern changes momentarily before returning to the same pattern.

The programmable onboard LED (marked “IO13”) is also connected to Pin 13, available on one of the pins of the male header (marked “D13”). As a further test, the positive terminal of a standard LED can be put onto the male header Pin 13, with the negative terminal of the LED attached to a resistor (in the range of ~200 Ω to 1000 Ω, give or take). The other free end of the resistor can be plugged onto one of the ground (marked “GND”) pins in a male header. The LED and resistor can either be soldered together, or a breadboard can be used to make the electrical connection.

6.5.2 Test 2: Onboard LED (Pin 13) Blink Reset Pattern

The reset button triggers a momentarily different pattern consisting of four rapid blinks of the onboard LED (marked “13”) of 50 mS ON and 50 mS OFF. This tests that the *Leonardo+* is restarting properly.

Upon reset the *Leonardo+* output several messages to the Serial monitor on baud rate 9600. To see these messages, load the IDE software and plug in the *Leonardo+*. Turn on the Serial monitor (select baud rate 9600) and read the output window.

6.5.3 Test 3: I2C Scanner

Every five seconds the I2C bus (SDA and SCL) is scanned for any connected peripherals. The I2C bus is available on the male header pins (See Section 3.5**Error! Reference source not found.** for the pinout diagram). The results of the scan are printed to the Serial monitor on baud rate 9600. To see these results, load the IDE software and plug in the *Leonardo+.* Turn on the Serial monitor (select baud rate 9600) and read the output window.

The user can connect several I2C peripherals at once. All the device addresses will be displayed in the Serial monitor.

6.5.4 Test 4: Pin 9 Fade

Pin 9 available on the male header (marked “D9”) is a PWM capable pin. PWM allows a pin to be driven at different duty cycles. This, among many other examples, can be used to dim, or fade, an external LED. In a similar fashion as outlined in Test 1, connect an LED plus resistor to Pin 9 and ground (marked “GND”) and observe the LED fading pattern. Ensure that the LED positive terminal is in Pin 9 and that the negative terminal goes toward GND through the resistor.

6.5.5 Test 5: Analog A0 Read

The analog pin A0 is programmed to be continuously reading any input connected to it. The read input value is displayed in the Serial monitor on baud rate 9600. To see these results, load the IDE software and plug in the *Leonardo+*. Turn on the Serial monitor (select baud rate 9600) and read the output window.

The user can plug a wire directly onto the male header pin A0 and the other end onto:

- A0 to Ground (marked “GND”)
- A0 to 3.3 V
- A0 to 5 V
- A0 free floating

The output as displayed in the Serial monitor will read different values accordingly. A properly working *Leonardo+* should produce the following results:

- A0 to Ground (marked “GND”) -> Output around 0
- A0 to 3.3 V -> Output around 660, plus or minor a few
- A0 to 5 V -> Output around 1023, plus or minor a few
- A0 free floating -> Output ranges widely

7 REFERENCES

This section lists relevant references.

- ATmega32U4 datasheet by Microchip Technology:
  https://ww1.microchip.com/downloads/en/devicedoc/atmel-7766-8-bit-avr-atmega16u4-32u4_datasheet.pdf

- PTSolns website:
  https://PTSolns.com

- PTSolns Documentation Repository:
  https://docs.PTSolns.com

- *PTSolns IDE* software:
  https://PTSolns.com/IDE

- Arduino IDE software:
  https://www.arduino.cc/en/software

- Sparkfun’s QWIIC Standard:
  https://www.sparkfun.com/qwiic

- PTSolns support:
  https://ptsolns.com/contact-us

Copyright © 2026 Pylon Technology Solutions Ltd.