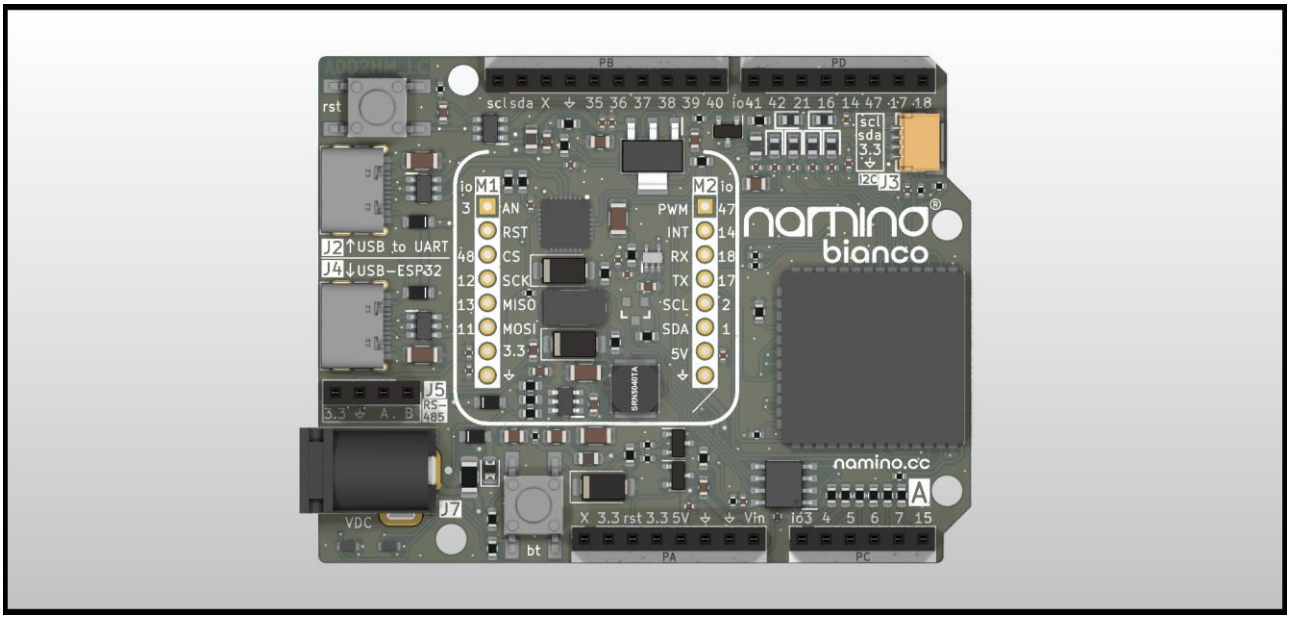
BIANCO

User manual


PB
PD
rst
scl sda X 35 36 37 38 39 40 io41 42 21 16 14 47 17 18
scl
sda
3.3
I2C J3
M1
io
3 AN
RST
48 CS
12 SCK
13 MISO
11 MOSI
3.3
M2
io
PWM 47
INT 14
RX 18
TX 17
SCL 2
SDA 1
5V
J2 ↑USB to UART
J4 ↓USB-ESP32
J5
3.3 A. B
RS-485
namino®
bianco
namino.cc
A
J7
VDC
bt
X 3.3 rst 3.3 5V Vin io3 4 5 6 7 15
PA
PC

NAMINO

1. Introduction

1.1. Reference standards

The **NAMINO** device is designed and manufactured in compliance with current legislation for the CE marking for devices that can be used in industrial and residential environments according to the:

EN 55032:2015 /AC:2016 /A11:2020 /A1:2020

EN 55035:2017 /A1:2020

2. Product description

Namino BIANCO is a basic device able to connect other devices and Arduino Uno shields. One other advantage is that Namino BIANCO is Open Hardware, so you can see how it is made.

Something about Namino:

Programming can be done by means of Arduino/ESP32 and VS code (PlatformIO) tools. See Software section.

Each board has a connector compatible with the Arduino Uno shields.

Extremely compact form-factor.

The widest range of possible power supplies: from 5Vdc÷28Vdc.

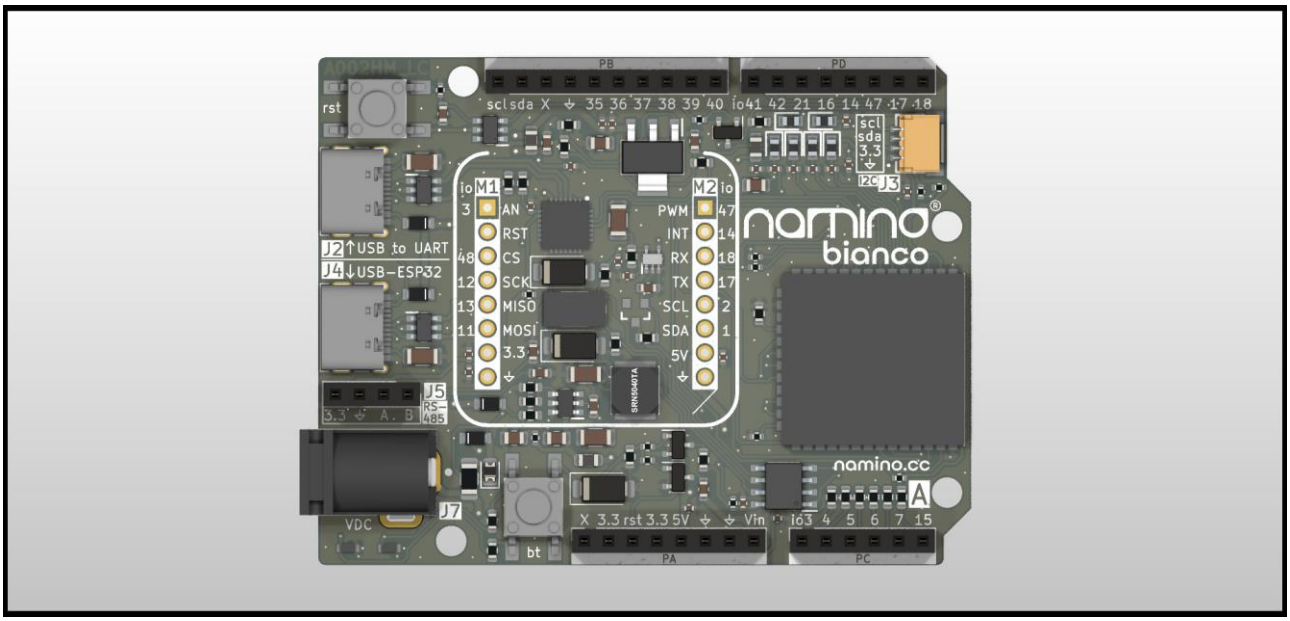
In summary:

Namino BIANCO can be used in the most challenging application and its special features can simplify interfacing with other devices.

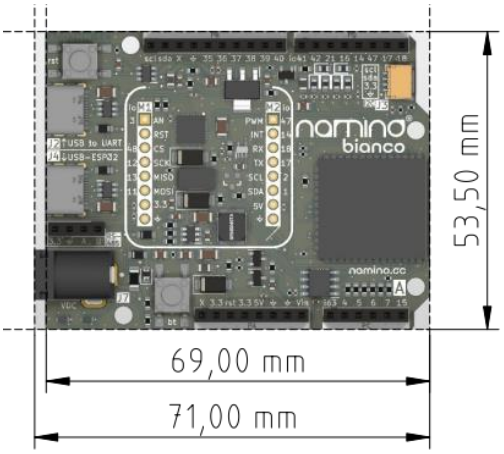
Sensors and devices belonging to both the industrial and amateur world can be easily used for the development of new ideas.

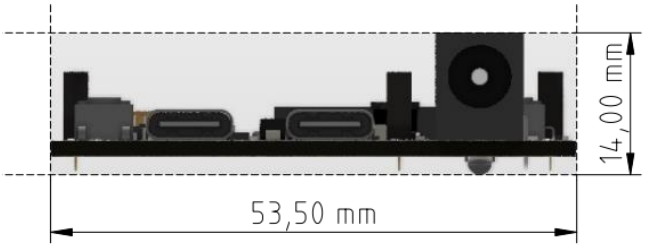
Namino boards has been designed and built directly by Mect Srl: an Italian manufacturer of PLC and products for industrial automation with over 40 years of history.

The following images show the board BIANCO:



2.1. Dimensions





2.2. Generic power supplies

NAMINO can be powered by various sources:

- 5Vdc via USB
- 7Vdc ÷ 28Vdc

3. Hardware

3.1. CPUs and memories

- ESP32-S3-WROOM-1U-N4R8 (ESP32-S3 series of embedded SoCs, Xtensa®dual-core 32-bit LX7 microprocessor, up to 240MHz)
- 4MB Flash
- 512KB SRAM
- 8MB PSRAM

3.2. Inputs / outputs

Namino BIANCO provides the following input and output lines:

- Arduino Uno shield interface
- n. 6 analog input capable of handling signals up to 0-30V (see below)
- Mikroe interface
- Qwiic interface
- 2 x USB type C device for programming and usage (NOT OTG)
- 1 pushbutton for programming
- 1 pushbutton for reset
- 1 led white programmable
- 1 led white as power indicator
- RS485 (for modbus RTU master/slave as well)
- Wi-Fi (802.11 b/g/n, Bit rate: 802.11n up to 150 Mbps) - need antenna (not included)
- Bluetooth LE: Bluetooth 5, Bluetooth mesh - need antenna (not included)
- Wi-fi / bluetooth antenna connector

NOTE: WiFi and BT shares the same antenna.

4. Boards features

The BIANCO board has a form factor that allows it to be used in various areas and allows it to be inserted into various commercially available enclosures.

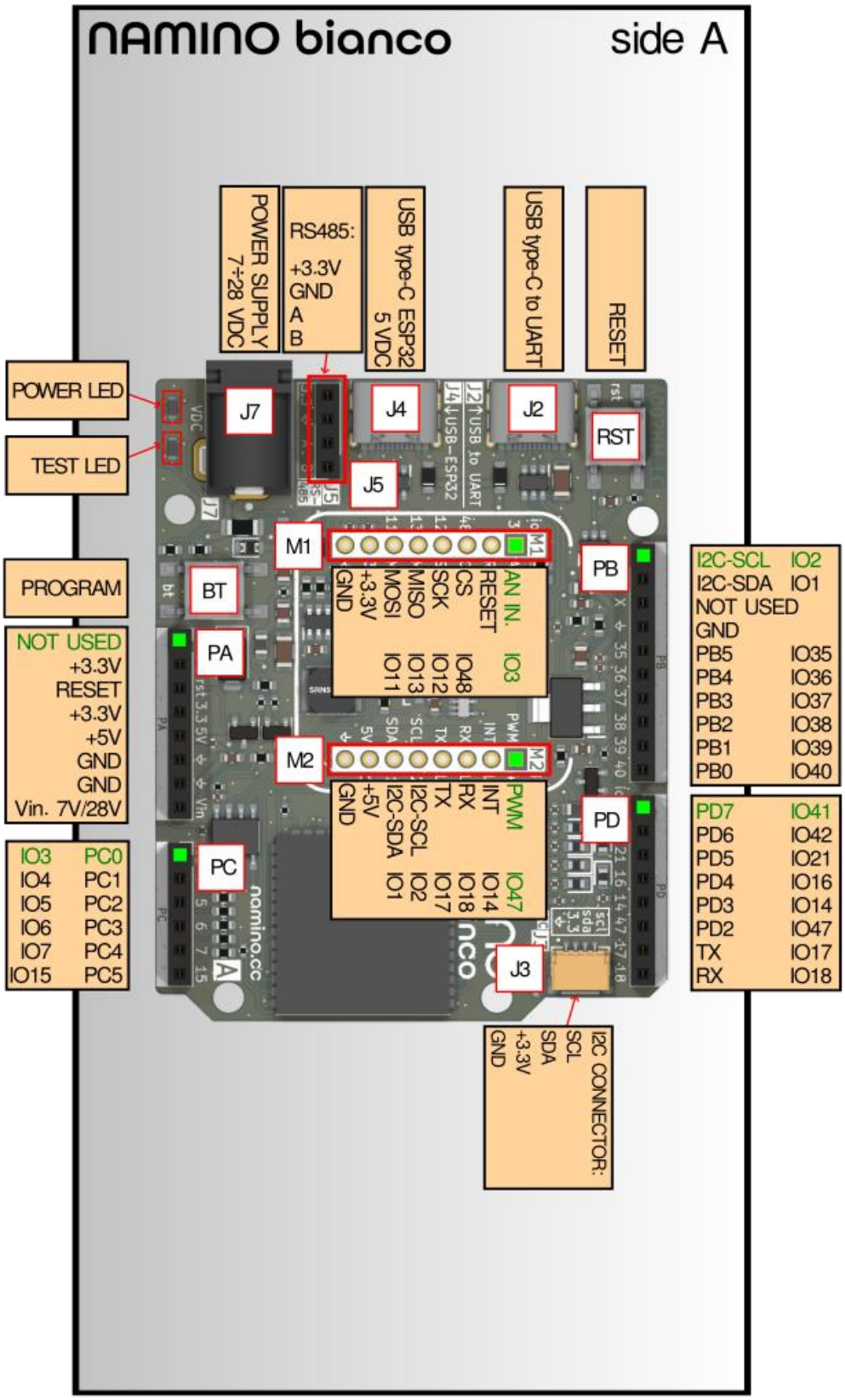
The wide power supply and configurable inputs via a divider allows the input voltage of each individual input pin to be configured directly on its PCB. There is also space to insert a filter capacitor directly on the PCB, allowing cleaner and more reliable wiring.

The BIANCO board also offers a wide variety of interfaces for the use of specific shields, in particular allowing the use of Arduino Uno, Mikroe and Qwiic shields.

It also provides two USB type C interfaces and also an RS485 interface with which it is possible to interface ModBus devices typical of the industrial world.

4.1. BIANCO Pinout

The following figure shows the pinout of the board:



Left side		Right side			
ARDUINO pin name	**ESP32 pin name**	**ARDUINO pin name**			**ESP32 pin name**
		SCL			SCL-IO0
	3V3	SDA			SDA-IO1
RESET	GRESET	AREF			X
3V3	3V3	GND			
5V	5V	PB5	SCK		IO35
GND	GND	PB4	MISO		IO36
GND	GND	PB3	MOSI	PWM	IO37
+VDC	VIN	PB2	SS	PWM	IO38
PC0	IO3_ADC1_CH2	PB1		PWM	IO39
PC1	IO4_ADC1_CH3	PB0	CP1		IO40
PC2	IO5_ADC1_CH4	PD7	ANIN1		IO41
PC3	IO6_ADC1_CH5	PD6	ANIN0		IO42
PC4	IO7_ADC1_CH6	PD5	T1		IO21
PC5	IO15_ADC2_CH4	PD4	T0		IO16
		PD3	INT1		IO14
		PD2	INT0		IO47
		PD1	TX		IO17_TX
		PD0	RX		IO18_RX

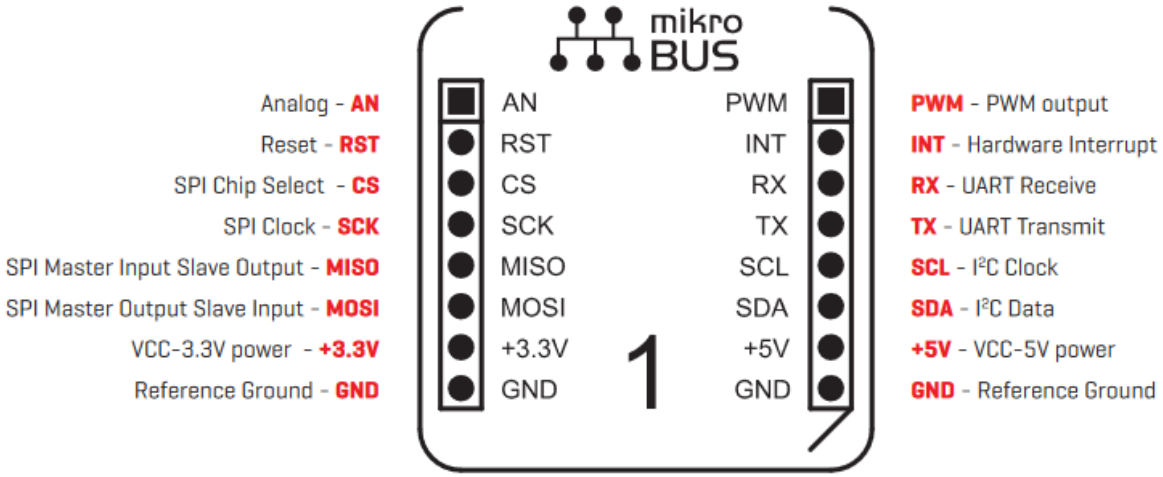
Associations that are different from the RED and ORANGE cards are highlighted in yellow

4.1.1. Programmable led

TEST LED is connected to ESP32 pin **IO47**.

4.1.2. Mikroe interface

BIANCO implements mikroBUS interface. The picture below shows the pinout.



The mikro BUS ESP32 pinout association is realized according to the following table:

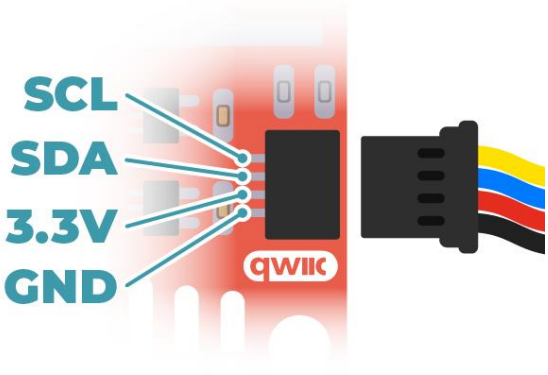
Left side		**Right side**	
Mikroe pin name	**ESP32 pin name**	**Mikroe pin name**	**ESP32 pin name**
AN	IO3_ADC1_CH2	PWM	IO47
RST	GRESET	INT	IO14
CS	IO48	RX	TX_IO17
SCK	IO12	TX	RX_IO18
MISO	IO13	SCL	SCL
MOSI	IO11	SDA	SDA
3V3	3V3	+5V	+5V
GND	GND	GND	GND

4.1.3. UART TTL interface

UART0 is made available for debugging via the second USB and the USB-UART converter.

4.1.4. Qwiic interface

The board implements the Qwiic connector. The connector exposes the I2C interface so the SDA and SCL lines already present on the other connectors will also be shared on the Qwiic compatible connector.



4.1.5. RS485 interface

J5 connector	Signal name
1	D-
2	D+
3	GND
4	3.3V

The BIANCO board also provides an RS485 serial interface.

4.1.6. USB1 ESP32 interface

This interface is used for programming the device.

4.1.7. USB2 ESP32 uart

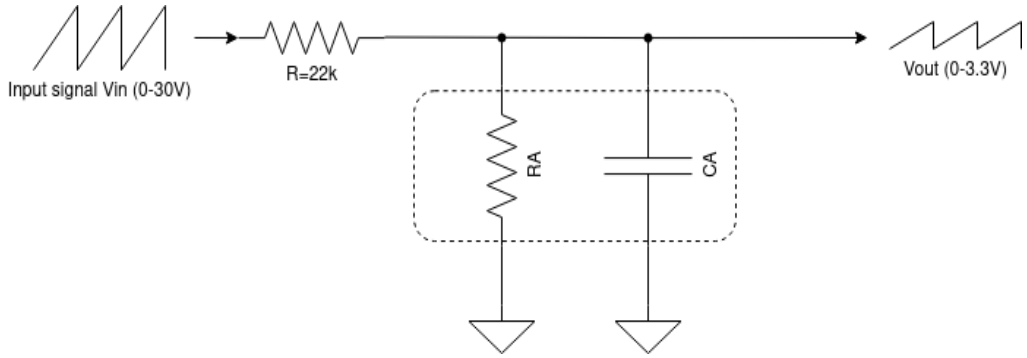
There is a second USB type C connector on the board, which communicates with ESP32 on UART0 via the USB serial converter.

4.1.8. Analog input divider calculation

***IMPORTANT**: **PC port is considered mainly as analog input**, because it has a 22kΩ in series, so if you want to use it as digital output you must go into an high impedance input logic and use it as slow logic signal. Please don't consider these pins for example, to drive directly a LED.*

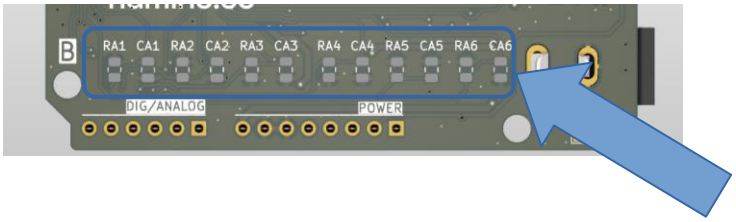
The white board on each solder-side analogue input has pads for soldering a resistor to adjust the maximum input voltage. The resistors must be **SMD 0805** type.

The input circuit for analog input is schematized in the following:



Components surrounded by the dashed rectangle are on the bottom side of the board (see below). The user can solder the resistor (R_A) in order to reduce input voltage. In addition there is the place for a capacitor (C_A) for a simple passive low pass filter.

There are *RA*, *CA* for each analog inputs and each channel can be different.



*NOTE: **RAx** and **CAx**, where **x** goes from 1 to 6 indicates resistors and capacitors that can be set and soldered by the user. In the following formulas **RAx** is **R_A** and **CAx** is **C_A**.*

IMPORTANT: the voltage range of these inputs must be between 0 – 30 V as maximum. All others inputs must be in the range of 0 – 3.3V.

If you need to reduce the input voltage, you have to solder only R_A resistor for the channel used. Use the following formulas to calculate their value. Suppose we have a maximum input voltage V_{inmax}, the resistor to be soldered is calculated as follows:

$$K = \frac{V_{outmax}}{V_{inmax}} = \frac{3.3}{V_{inmax}}$$

then use it in this:

$$R_A = \frac{R \cdot K}{1 - K} = \frac{22000 \cdot K}{1 - K}$$

R_A will be the resistor value to be soldered on the pad correspondent to the desired analog channel.

For example:

suppose you have an analogue input that goes from 0 to 12 V, you will have to partition the signal to get it to 3.3V so:

$$K = \frac{V_{outmax}}{V_{inmax}} = \frac{3.3}{12} = 0.275$$

$$R_A = \frac{R \cdot K}{1 - K} = \frac{22000 \cdot 0.275}{1 - 0.275} \simeq 8345\,\Omega$$

The nearer best commercial value is 8.2 kΩ.

Let's go to the verification:

$$V_{out} = V_{inmax} \cdot \frac{R_A}{R + R_A} = 12 \cdot \frac{8200}{22000 + 8200} \simeq 3.26\,V$$

Value is less than 3.3V so it is acceptable.

If you need an input filter, you can solder a capacitor C_A for the channel used.

It will be a simple first order Low Pass filter (RC). You can calculate it by the following formula for the

$$f = \frac{1}{2\pi R C_A}$$

cutting frequency:

From this formula, if you know the frequency you can invert it to calculate the capacitor.

If you need to reduce the input voltage and you need filtering, you can solder a resistor and a capacitor for the considered analog channel. Formulas to be used are the following:

$$V_{out} = \frac{R}{R + R_A} V_{inmax}$$

$$f = \frac{1}{2\pi \cdot R \, R_A \cdot C_A}$$

If you know the frequency you can invert the above formula to calculate C_A.

5. Programming

IMPORTANT: Boards PCB with code "A002HM" can be programmed only using a USB type A → USB type C cable.

5.1. Introduction

NAMINO can be programmed in C/C++ using::

- Arduino IDE
- Platformio.org

Note: press the **BT button** to enter in programming mode.

NOTE: Please follow *Namino forum* and *Namino Youtube channel* to see tutorials on installing and deploy.

To see how to program Namino BIANCO, please refer to examples.

The header of the file contains the version/date:

```
Copyright (c) 2023 Namino Team, version: x.y.z @ yyyy-mm-dd
```

5.1.1. File platformio.ini

Use the following code to initialize NAMINO profile with platformio, some code is useful in future for console/debug/etc:

```
[env:adafruit_feather_esp32s3_reversetft]
platform = espressif32
board = adafruit_feather_esp32s3_reversetft
board_build.f_cpu = 240000000L
framework = arduino
monitor_speed = 115200
monitor_echo = yes
monitor_eol = LF
monitor_filters =
  colorize
  time
  send_on_enter
```

5.1.2. Initial setup

Use the following code to initialize the NAMINO BIANCO program:

```
#include <Arduino.h>
#include "./namino_bianco_pins.h"

[put here your code]
```

5.2. Examples

Examples can be downloaded from the Github section.