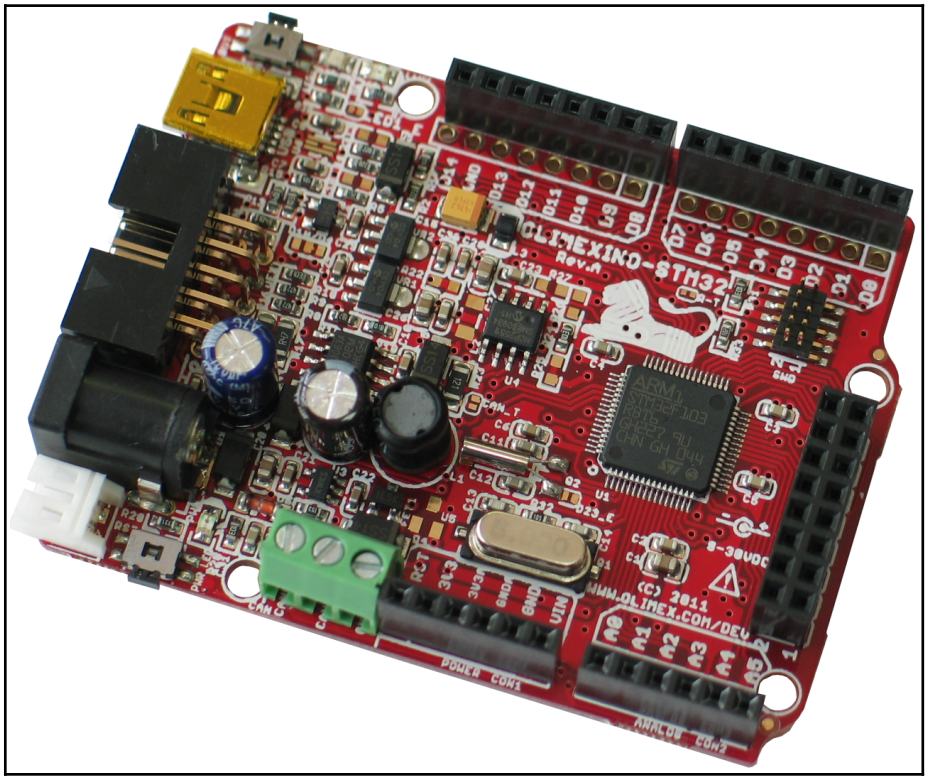


OLIMEXINO-STM32 development board

User's manual

lead-free

All boards manufactured by Olimex are ROHS compliant

Document revision H, May 2025

Designed by OLIMEX Ltd, 2012

INTRODUCTION TO DUINO:

Arduino is an open-source electronics prototyping platform, designed to make the process of using electronics in multidisciplinary projects easily accessible. The hardware consists of a simple open hardware design for the Arduino board with an Atmel AVR processor and on-board I/O support. The software consists of a standard programming language and the boot loader that runs on the board.

Arduino hardware is programmed using a Wiring-based language (syntax + libraries), similar to C++ with some simplifications and modifications, and a Processing-based Integrated Development Environment (IDE). The project began in Ivrea, Italy in 2005 aiming to make a device for controlling student-built interaction design projects less expensively than other prototyping systems available at the time. As of February 2010 more than 120,000 Arduino boards had been shipped. Founders Massimo Banzi and David Cuartielles named the project after a local bar named “Arduino”. The name is an Italian masculine first name, meaning "strong friend". The English pronunciation is "Hardwin", a namesake of Arduino of Ivrea.

More information could be found at the creators web page http://arduino.cc/ and in the Arduino Wiki http://en.wikipedia.org/wiki/Arduino

To make the story short – Arduino is easy for beginners who lack Electronics knowledge, but also does not restrict professionals as they can program it in C++ or mix of Arduino/C++ language. There are thousands of projects which makes it easy to startup as there is barely no field where Arduino enthusiasts to have not been already.

Arduino has inspired two other major derivatives – MAPLE and PINGUINO. Based on 8-bit AVR technology the computational power of Arduino boards is modest, this is why a team from MIT developed the MAPLE project which is based on ARM7 STM32F103RBT6 microcontroller. The board has same friendly IDE as Arduino and offers the same capabilities as hardware and software but runs the Arduino code much faster. Unfortunately, by year 2016, the MAPLE project is now no longer supported and updated. The remaining resources of the Maple project can be found at https://www.leaflabs.com/maple

In parallel with Arduino another project was started called PINGUINO. This project chose its first implementation to be with PIC microcontrollers, as AVRs were hard to find in some parts of the world like South America so it is likely to see lot of PINGUINO developers are from that part of the world. PINGUINO project founders decided to go with Python instead Java for processing language. For the moment PINGUINO is much more flexible than Arduino as it is not limited to 8bit microcontrollers. Currently the IDE, which has GCC in background, can support 8-bit PIC microcontrollers, 32bit PIC32 (MIPS) microcontrollers and ARM7/CORTEXM3 microcontrollers which makes PINGUINO very flexible because once you make your project you can migrate easily through different hardware platforms and not being bound to a single microcontroller manufacturer. The PINGUINO project can be found at: http://www.pinguino.cc.

BOARD FEATURES:

We entered the Arduino/MAPLE field 5 years after the design was introduced, and this allowed us to see and resolve some of (what we consider) errors made by the Arduino inventors.

We had the possibility to read current customer feedback and to implement what they wanted to see in the original Arduino.

1. Original Arduino/MAPLE uses linear power supply, this limits the input voltage range. We designed the power supply to accept power from 9 to 30V DC thus making it possible to take virtually any power supply adapter on the market, also enable application which are in industrial power supply 24VDC;
2. We carefully selected all components to work reliable in **INDUSTIRAL temperature range -25+85C** so the board can be used in INDUSTIRAL applications while the original design is to Commercial 0 – 70C operating temperature;
3. The original Arduino/MAPLE design is not very reliable for portable applications as consumes it too much power with the linear voltage regulators, we put ULTRA LOW POWER voltage regulators and the consumption is only few microampers, which enables handheld and battery powered applications;
4. We add Li-Ion rechargable battery power supply option with BUILD-IN on board charger, so when you attach battery it is automatically charged and kept in this state until the other power source (USB or external adapter) is removed and it AUTOMATICALLY will power the board – no jumpers, no switches;
5. Our board has UEXT connector which allows many existing modules like RF, ZIGBEE, GSM, GPS to be connected;
6. Our board has micro SD card;
7. Our board has CAN driver on board;
8. Our design allows RTC – Real Time Clock;
9. We made our design noise immune;
10. We use separate voltage regulator for the Analog part, which allows the ADC to be read correctly without the digital noise pickup;
11. The LEDs and the BUTTONs are on the edge of the board so there is easy access even if the boards have shields on them;
12. All components are LOWER than the connectors, so the shields do not interfere with them;
13. mini USB connector is used which is common and used in most cell phones, so you do not have to buy other cables;
14. Original Arduino design had a flaw and the connectors were not spaced at 0.1" thus making; breadboarding board use impossible, to keep the compatibility we have the same spacing but we added next to this connector on 0.1" which customer can use with perforated boards;
15. All signals on the connectors are printed on top and on bottom of the board, so when you check with probe you know exactly which port you are measuring;
16. 4 mount holes make board attachment easier.

ELECTROSTATIC WARNING:

The **OLIMEXINO-STM32** board is shipped in protective anti-static packaging. The board must not be subject to high electrostatic potentials. General practice for working with static sensitive devices should be applied when working with this board.

BOARD USE REQUIREMENTS:

HARDWARE REQUIREMENTS:

- Cables: You'll need a USB cable with mini USB connector for connecting the board to a personal computer and Arduino/MAPLE IDE. The board also gets powered via this cable.

- USB-serial adapter (optional but highly recommended): It should have free leads and 3.3V voltage levels, it will allow you to program the board via the UART at the UEXT connector in case the bootloader gets corrupted. Consider this cable:

https://www.olimex.com/Products/USB-Modules/Interfaces/USB-SERIAL-F/

- ARM programming tool (optional but recommended): STM32-compatible programmer/debugger is recommended:

1. if you wish to use a general-purpose approach of programming and debugging (not USB nor USB-serial method);

2. if you wish to change the bootloader with another bootloader (also possible via USB-serial);

3. if you managed to overwrite the bootloader program and wish to recover the device (also possible via USB-serial).

You can use any STM32F103 compatible programmer or debugger just notice that the board has a 10-pin mini JTAG/SWD connector (0.05" step) – make sure that you have a suitable adapter. You can use ARM-JTAG-20-10 adapter for this task.

Our ARM-USB-OCD-H, ARM-USB-TINY-H tools are capable of programming the board (remember the 20-10 adapter).

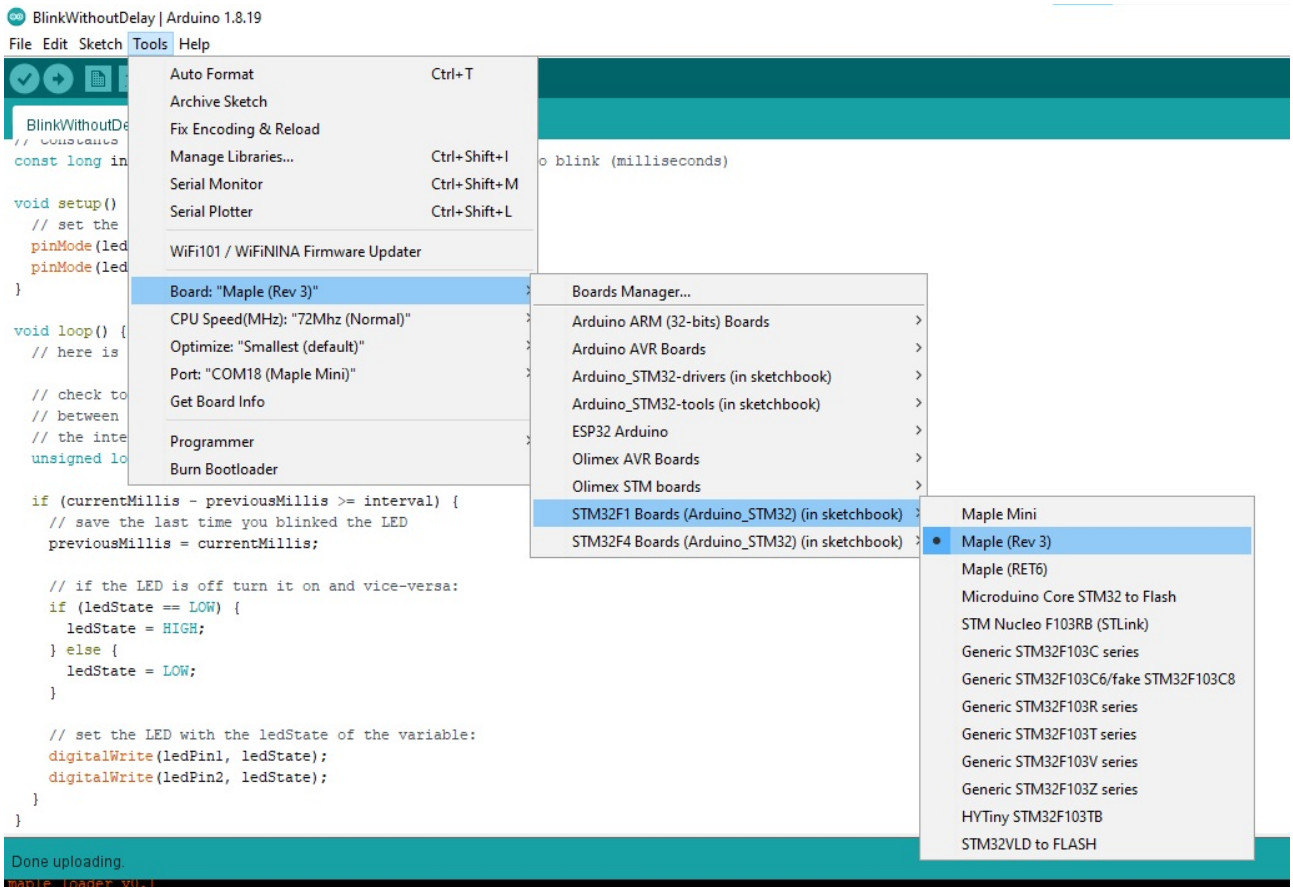
SOFTWARE REQUIREMENTS:

- Environment (recommended)

By default the board comes with Maple (rev 3) bootloader. It is recommended to use MAPLE IDE initially but since the MAPLE project is now completely abandoned it is recommended to either use an alternative (MPIDE; Espruino IDE; stm32duino; etc) or use general-purpose method of programming. Some environments or packages have built-in support for Maple (rev3) bootloader. If you want to use Arduino IDE refer to this guide:

https://github.com/rogerclarkmelbourne/Arduino_STM32/wiki/Installation

In board selection go to Tools → Board - > STM32F1 Boards → Maple (Rev 3). Refer to the picture below:



Select the port where the board is listed (remember to install drivers from archive if it is not listed) and load a demo. Here is a blink demo you can use as first demo:

// constants won't change. Used here to set pin numbers:

const int ledPin1 = PA1;// the number of the LED pin

const int ledPin2 = PA5;// the number of the LED pin

// Variables will change:

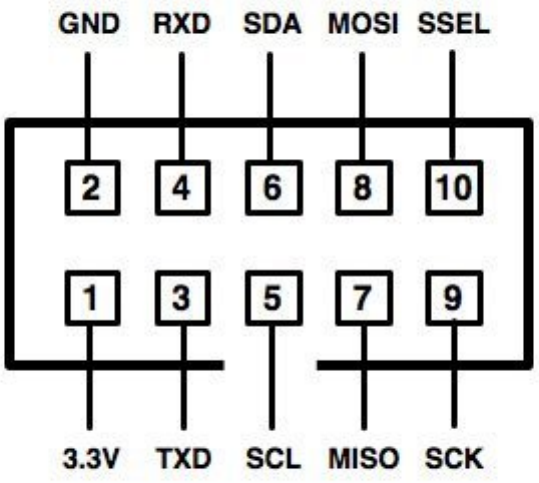
int ledState = LOW; // ledState used to set the LEDs

// Generally, you should use "unsigned long" for variables that hold time

```
// The value will quickly become too large for an int to store
unsigned long previousMillis = 0;        // will store last time LED was updated

// constants won't change:
const long interval = 1000;           // interval at which to blink (milliseconds)

void setup() {
  // set the digital pins as output:
  pinMode(ledPin1, OUTPUT);
  pinMode(ledPin2, OUTPUT);
}

void loop() {
  // here is where you'd put code that needs to be running all the time.

  // check to see if it's time to blink the LED; that is, if the difference
  // between the current time and last time you blinked the LED is bigger than
  // the interval at which you want to blink the LED.
  unsigned long currentMillis = millis();

  if (currentMillis - previousMillis >= interval) {
    // save the last time you blinked the LED
    previousMillis = currentMillis;

    // if the LED is off turn it on and vice-versa:
    if (ledState == LOW) {
      ledState = HIGH;
    } else {
      ledState = LOW;
    }

    // set the LEDs with the ledState of the variable:
    digitalWrite(ledPin1, ledState);
    digitalWrite(ledPin2, ledState);
  }
}
```

- Environment (recommended)

It is important to notice that the STM32 chip has internal serial bootloader which can be used to change any code without JTAG debugger (but it requires USB-serial cable since it is available at the UART routed to the UEXT). You can program the board via the on-board serial bootloader via USB-serial cable attached to the UEXT (GND of cable to GND of UEXT; RX of cable to TX of UEXT; TX of cable to RX of UEXT). Make sure to clearly identify the first and second pin of the UEXT connector to establish the proper hardware connection, the first pin has square pad on top, also refer to this layout picture of UEXT:



You can use USB-serial cable adapter, it is important its levels to be at 3.3V (not 5V) and also it is good idea to have free leads with female connectors for easier connection. Something like USB-SERIAL-F cable:

https://www.olimex.com/Products/USB-Modules/Interfaces/USB-SERIAL-F/

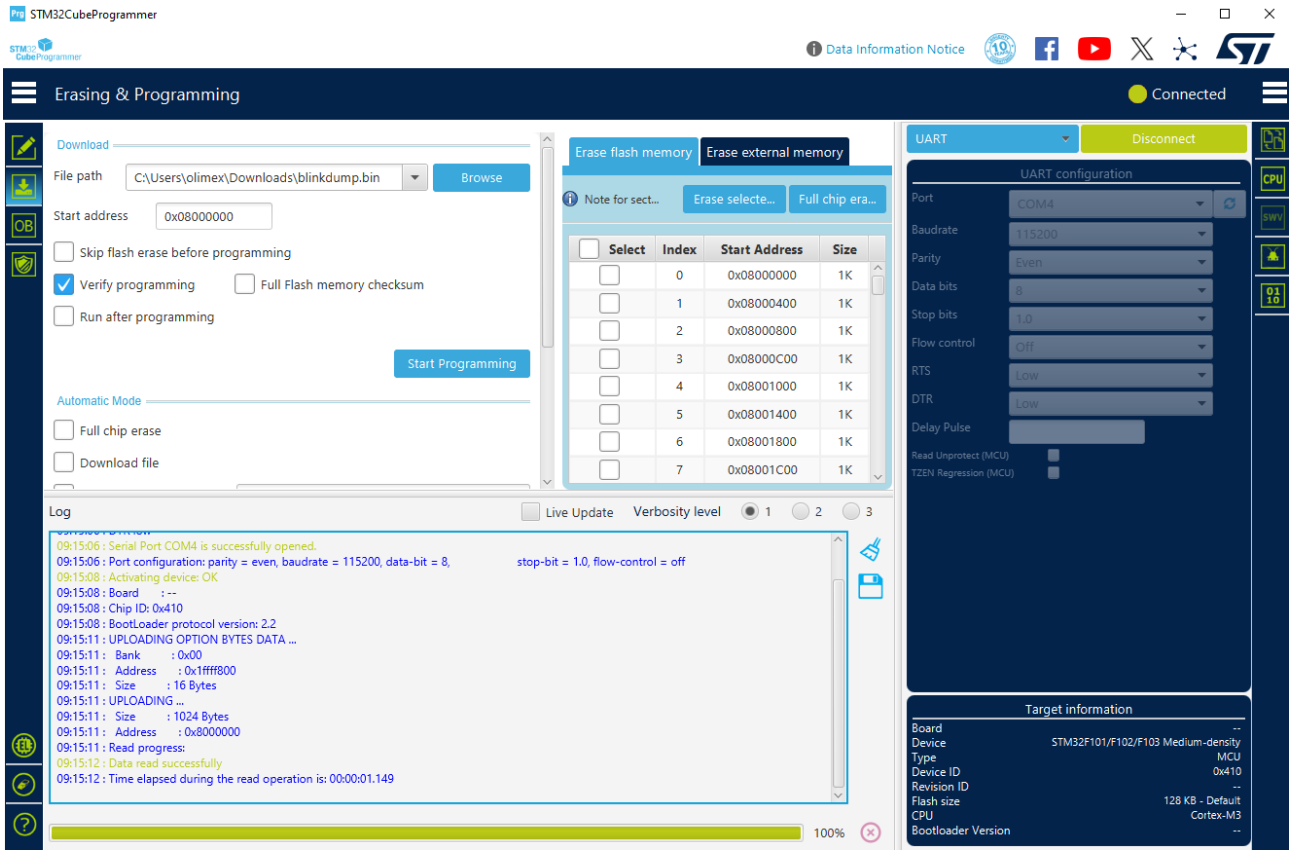
After the hardware connection between the USB-serial cable is established, you need to put the board in bootloader mode – this is done using the on-board buttons:

1. Press and hold button BUT

2. Press and release button PWR

3. Release button BUT

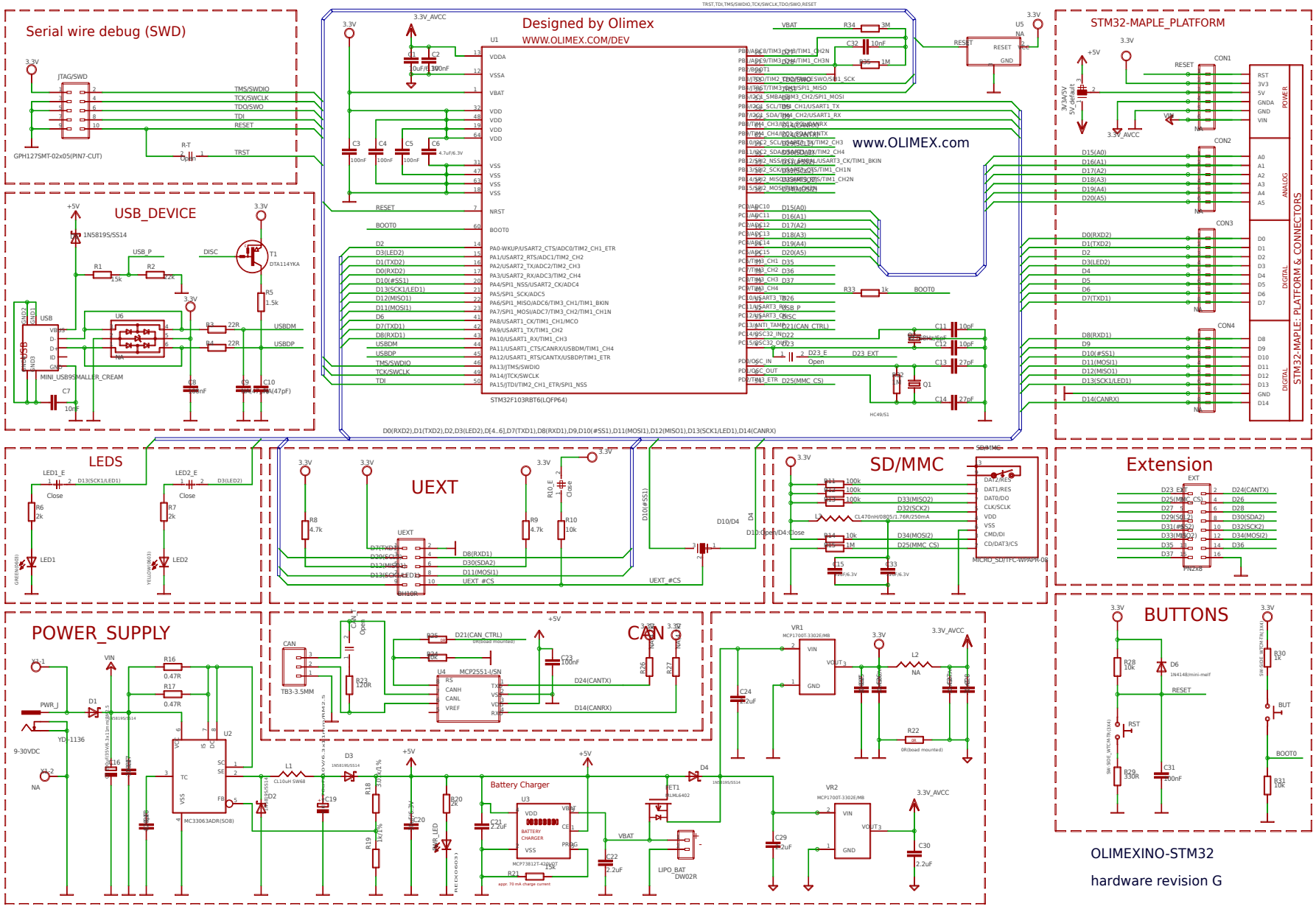
Now you can connect to the board via UART with tools like STM32CubeProgrammer or Arduino IDE or Espruino. There is a picture on the next page on how a successful connection looks in STM32CubeProgrammer:



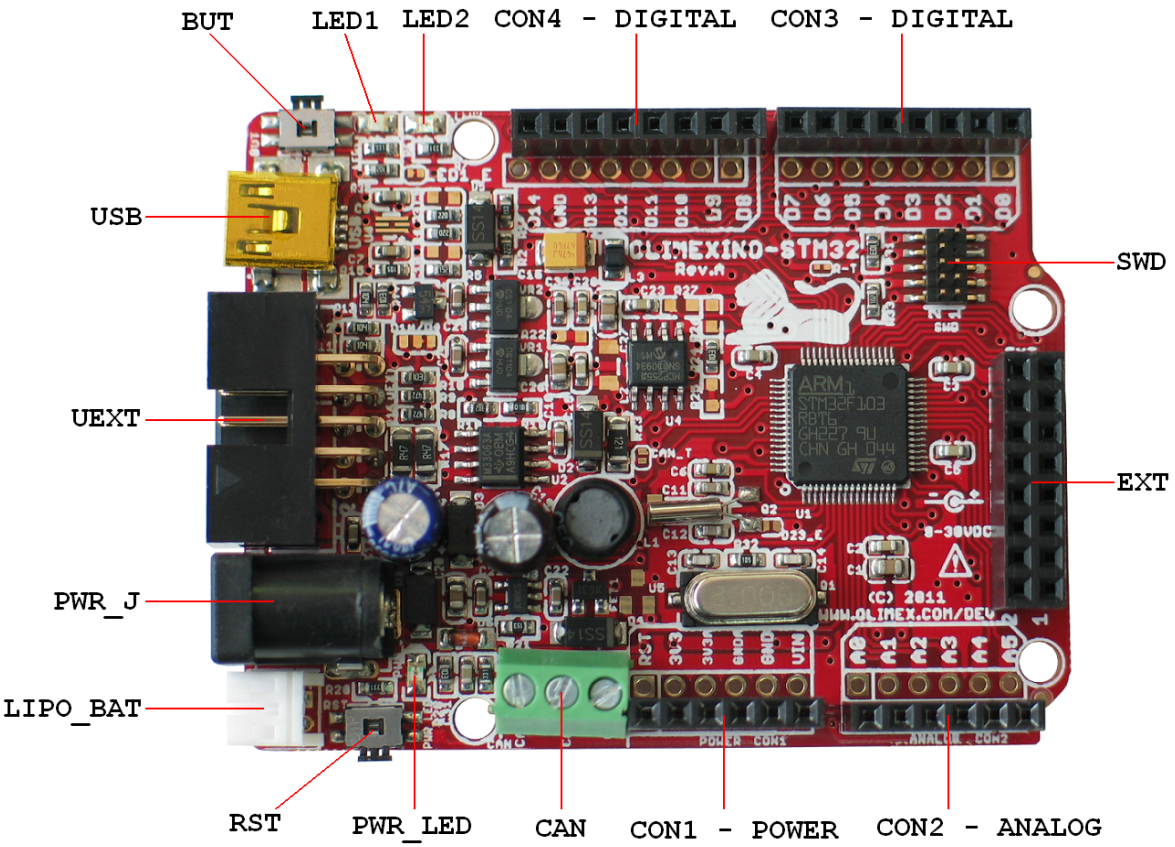
You can also find instructions on how to use the older flasher (that was used before STM32CubeProgrammer) “STM32 Flash Load Demonstrator” in the wiki article: https://www.olimex.com/wiki/How_to_use_OLIMEXINO-STM32_with_Espruino_IDE

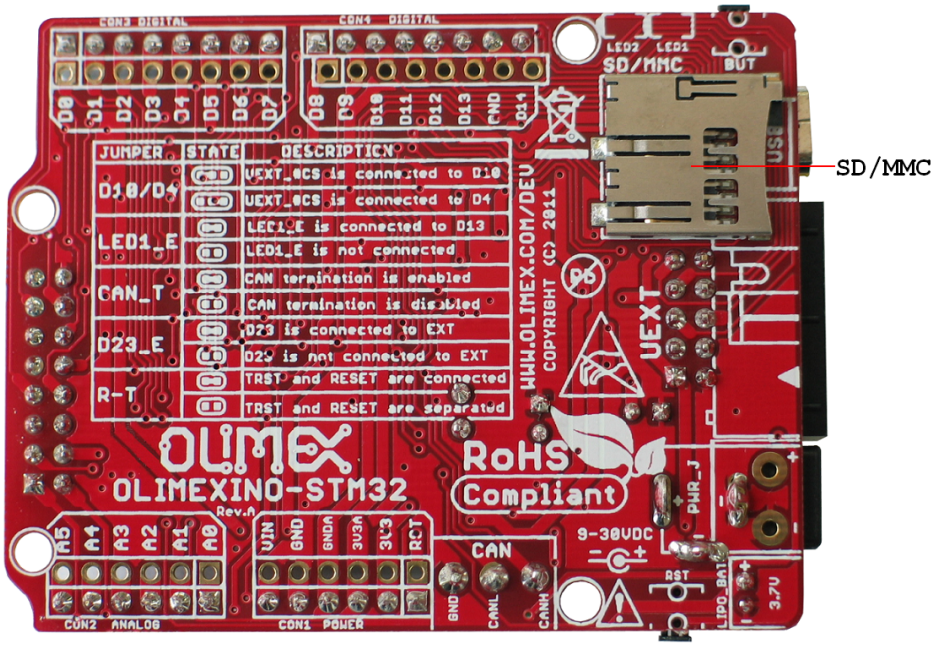
- The general-purpose method for programming and debugging require a JTAG/SWD programmer or debugger in order to interact with the board.

SCHEMATIC:


Serial wire debug (SWD)
Designed by Olimex
WWW.OLIMEX.COM/DEV
www.OLIMEX.com
STM32-MAPLE_PLATFORM
USB_DEVICE
LEDS
UEXT
SD/MMC
Extension
POWER_SUPPLY
CAN
BUTTONS
Battery Charger
STM32F103RBT6(LQFP64)
OLIMEXINO-STM32
hardware revision G

BOARD LAYOUT:





POWER SUPPLY CIRCUIT:

OLIMEXINO-STM32 can be powered from:

– external power supply (9-30) VDC.

– + 5V from USB

– 3.7 V Li-ion battery

– JTAG/SWD programmer/debugger

The programmed board power consumption is about 50 mA with all peripherals enabled.

RESET CIRCUIT:

OLIMEXINO-STM32 reset circuit includes D6 (1N4148), R28 (10kΩ), R29 (330Ω), C31 (100nF), STM32F103RBT6 pin 7 (NRST) and RESET button.

CLOCK CIRCUIT:

Quartz crystal **Q1** 8 MHz is connected to STM32F103RBT6 pin 5 (PD0/OSC_IN) and pin 6 (PD1/OSC_OUT).

Quartz crystal **Q2** 32.768 kHz is connected to STM32F103RBT6 pin 3 (PC14/OSC32_IN) and pin 4 (PC15/OSC32_OUT).

JUMPER DESCRIPTION:

Note that all jumpers on the board are SMD type. You will need to solder/unsoder/cut them in order to reconfigure them.

LED1_E

This jumper, whenclosed, enables LED1.

Default state is closed.

LED2_E

This jumper, when closed, enables LED2.

Default state is closed.

D23_E

This jumper, when is closed, connects STM32F103RBT6 pin (PC15/OSC32_OUT) – signal D23 and EXT pin 1, and when opened, D23 is not connected to EXT.
Default state is opened.

R-T

This jumper, when closed, connects TRST and RESET, and when opened, TRST and RESET are separated.
Default state is opened.

CAN_T

This jumper, when closed, enables CAN termination, and when opened, CAN termination is disabled.
Default state is opened.

D10/D4

This jumper, when set in position D10, UEXT pin 10 (UEXT_#CS) is connected to STM32F103RBT6 pin 20 (PA4/SPI1_NSS/USART2_CK/ADC4) – signal D10, and when put in position D4, UEXT pin 10 (UEXT_#CS) gets connected to STM32F103RBT6 pin 57 (PB5/I2C1_SMBAI/TIM3_CH2/SPI1_MOSI) – signal D4. Note that P10_E's effect also is affected by D10/D4.

Default state is in position D4.

P10_E:

When closed the board provides 3.3V to the UEXT_CCS – UEXT pin.

Default state is closed.

3V3A/5V

This is a new jumper introduced in hardware revision F of the board. It changes the voltage provided to pin #3 of the power shield connector from the default 5V to 3.3VA.

IMPORTANT! Hardware revision F introduced changes to the POWER connector of the Arduino shield. Now there are 5V available at pin #3 of the POWER connector by default!

It was 3.3V in older revisions. Be careful if you use 3.3V shields! You can revert the voltage of the pin back to 3.3V if you change the position of the 3V3A/5V jumper!

Default state is in position 5V.

INPUT/OUTPUT:

Status Led with name **LED1 (green)** connected via jumper LED1_E to STM32F103RBT6 pin 21 (PA5/SPI1_SCK/ADC5) – signal D13(SCK/LED1).

Status Led with name **LED2 (yellow)** connected to STM32F103RBT6 pin 15 (PA1/USART2_RTS/ADC1/TIM2_CH2) – signal D3(LED2).

Power-on LED (red) with name **PWR_LED** – this LED shows that the board is power supplied.

User button with name **BUT** connected to **STM32F103RBT6** pin 40 (PC9/TIM3_CH4) via R33 (1kΩ) and pin 60 (BOOT0) – signal BOOT0.

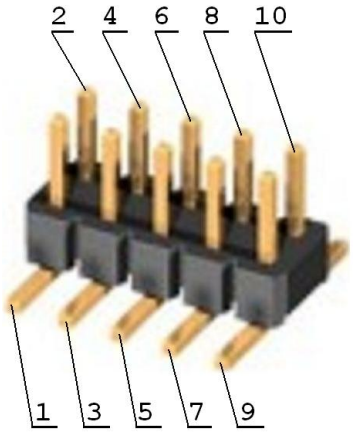
User button with name **RST** connected to STM32F103RBT6 pin 7 (NRST).

EXTERNAL CONNECTORS DESCRIPTION:

SWD:

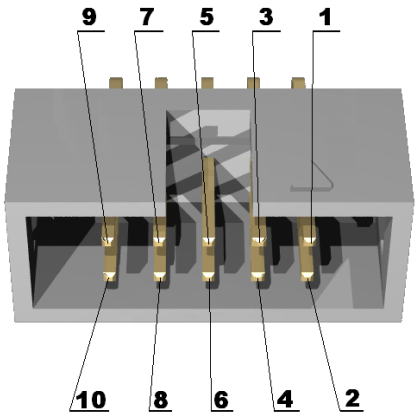
Pin #	Signal Name
1	VCC
2	TMS/SWDIO
3	GND
4	TCK/SWCLK
5	GND
6	TDO/SWO
7	Cut off
8	TDI
9	GND
10	RESET

Note that pin 7 of SWD connector is cut off.



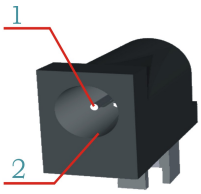
UEXT:

Pin #	Signal Name
1	VCC
2	GND
3	D7(TXD1)
4	D8(RXD1)
5	D29(SCL2)
6	D30(SDA2)
7	D12(MISO1)
8	D11(MOSI1)
9	D13(SCK/LED1)D13(SCK1/LED1)
10	UEXT_#CS



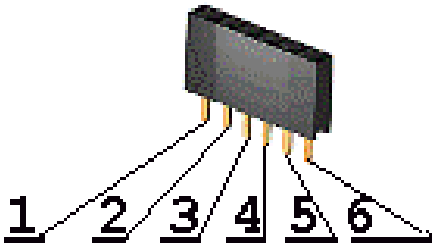
PWR_JACK:

Pin #	Signal Name
1	Power input
2	GND



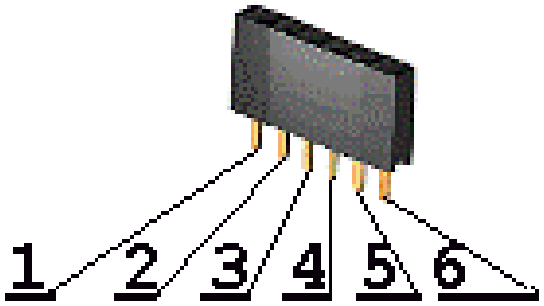
CON1 – POWER:

Pin #	Signal Name
1	RESET
2	3.3V OUTPUT
3	5V OUTPUT (3.3VA if you modify jumper 3V3A/5V)
4	GND
5	GND
6	VIN



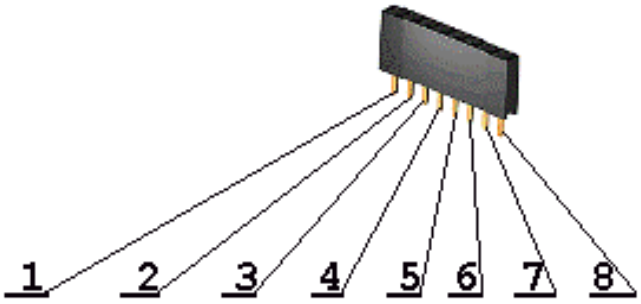
CON2 – ANALOG:

Pin #	Signal Name
1	D15(A0)
2	D16(A1)
3	D17(A2)
4	D18(A3)
5	D19(A4)
6	D20(A5)



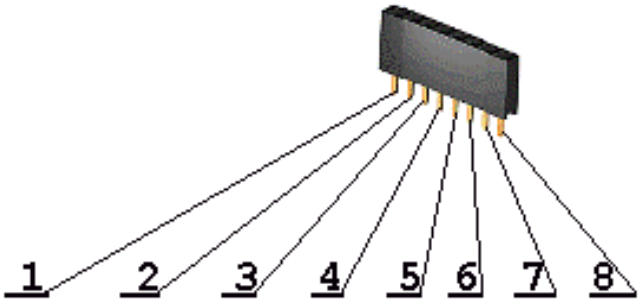
CON3 – DIGITAL:

Pin #	Signal Name
1	D0(RXD2)
2	D1(TXD2)
3	D2
4	D3(LED2)
5	D4
6	D5
7	D6
8	D7(TXD1)



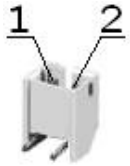
CON4 – DIGITAL:

Pin #	Signal Name
1	D8(RXD1)
2	D9
3	D10(#SS1)
4	D11(MOSI1)
5	D12(MISO1)
6	D13(SCK/LED1)
7	GND
8	D14(CANRX)



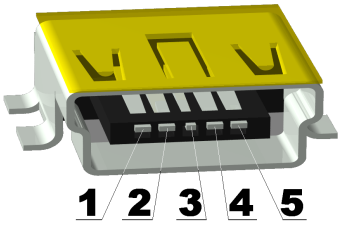
LI_BAT:

Pin #	Signal Name
1	VBAT
2	GND



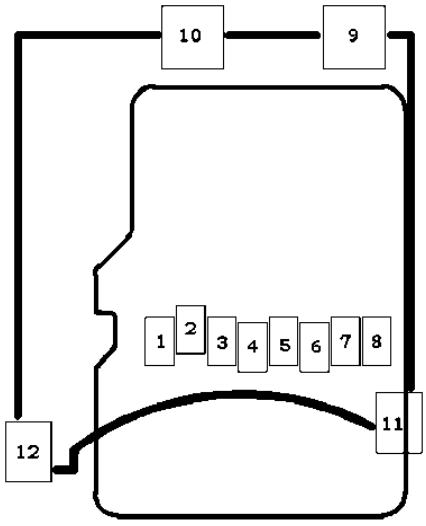
USB:

Pin #	Signal Name
1	+5V_USB
2	D -
3	D +
4	Not connected
5	GND

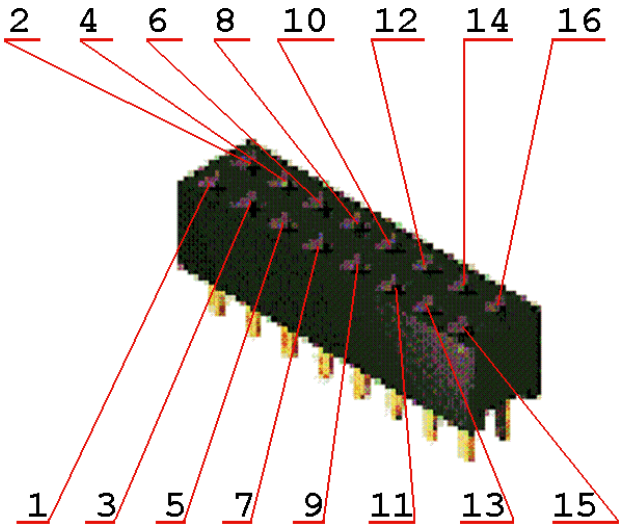


SD/MMC:

Pin #	Signal Name
1	MCIDAT2
2	D25(MMC_CS)
3	D34(MOSI2)
4	MMC_PWR
5	D32(SCK2)
6	GND
7	D33(MISO2)
8	MCIDAT1
9	Not connected
10	Not connected
11	Not connected
12	Not connected



EXT:

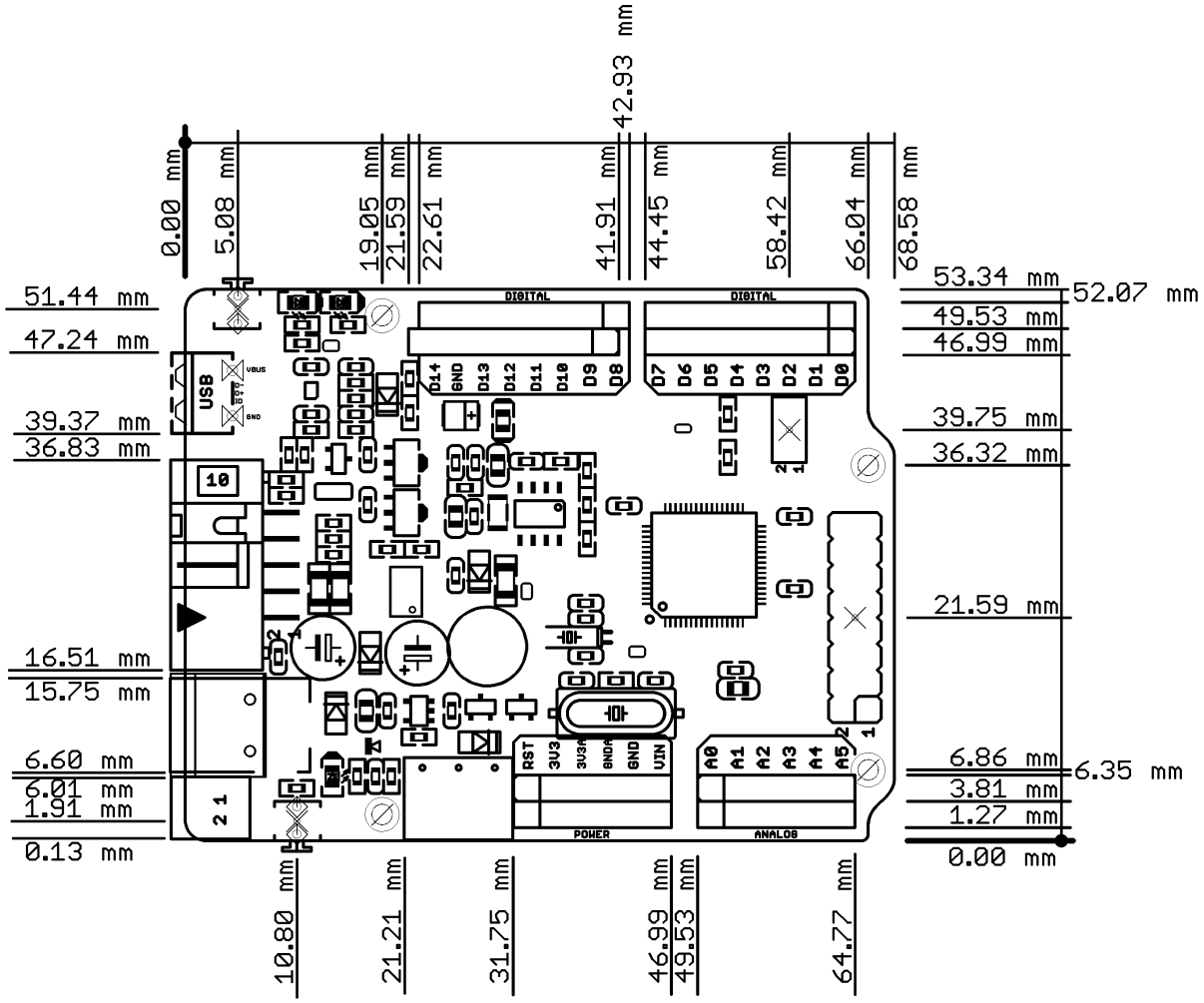


Pin #	Signal Name	Pin #	Signal Name
1	D23_EXT	2	D24(CANTX)
3	D25(MMC_CS)	4	D26
5	D27	6	D28
7	D29(SCL2)	8	D30(SDA2)
9	D31(#SS2)	10	D32(SCK2)
11	D33(MISO2)	12	D34(MOSI2)
13	D35	14	D36
15	D37	16	GND

CAN:

Pin #	Signal Name
1	GND
2	CANL
3	CANH

MECHANICAL DIMENSIONS:



AVAILABLE DEMO SOFTWARE:

The board comes with a simple program on-board. To get more projects, examples and ready maple libraries please visit the OLIMEXINO-STM32's page: https://www.olimex.com/Products/Duino/STM32/OLIMEXINO-STM32/

ORDER CODE:

OLIMEXINO-STM32 – assembled and tested board

How to order?

You can order directly from our web-shop or from any of our distributors.
Check our web https://www.olimex.com for more info.

Board revision history:

Revision	Notable Changes
A	- C6 (100n/0603) is changed to 4.7uF/0603.
B	- Removed the label "<c> 2011". - Logos added: Open Hardware, Designed by OLIMEX and Made in Bulgaria, 2011 logos - Added divider which includes R34, R35 and C32 with aim to measure the battery. - All tracks which were placed close to board's edges were moved as far as possible away from them. - Some changes in the values of some components were made
C	- Added closed by default SMD jumpers on LED2 and R10(UEXT_CS)lines - The table with the jumper description is now updated - Some logos and print lines have been re-arranged
D	- Added again the PWR JACK connector for the external supply in the board design and schematic
E	- R18 is changed from 3k/1% to 3.01k/1% - C15(tantalum 47uF/6.3V) is removed and replaced by 2 (two) 22uF/6.3V0603 capacitors
F	- Wires near the edges of the board were moved to the center - By default the board now provides 5V on the Arduino connector (at the 5V pin). Added 3-pin SMD jumper at the bottom to change to 3.3VA. - Different SWD connector used now - Placing new and better USB OTG connector - Placing new and better micro SD card connector
G	- Minor changes related to the USB placement and some labels

Document revision history:

Revision	Changes	Modified Page#
A	- At first page “Copyright(c) 2011, OLIMEX Ltd, All rights reserved” is replaces with “Designed by OLIMEX Ltd., 2011” - Updated schematic with board revision A	1
B	- Updated schematics with board revision B and table	6, 11
C	- Fixed grammatical errors - Updated schematics with board revision D - Added board revision history - Updated schematics - Added new jumpers description - Updated disclaimer	All
D	- Removed misleading feature – there is no precision AREF circuit on the board	4
E	- Schematic and revision changes updated to reflect latest board revision	5, 16
F	- Updated manual for hardware revision F – the most important change is the 5V output at pin #3 of the Arduino shield connector (it was 3.3VA previously). This can be reverted by modifying the 3V3A/5V jumper.	All
G	- Added a note about the changes to the output pin of the POWER connector	8
H	- Added info about Arduino IDE installation - Updated schematic to revision G	5, 6, 9

DISCLAIMER

The information in this document is provided in connection with Olimex products. No license, express or implied or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Olimex products.

The Hardware project is released under the Creative Commons Attribution-Share Alike 3.0 United States License. You may reproduce it for both your own personal use, and for commercial use. You will have to provide a link to the original creator of the project https://www.olimex.com on any documentation or website.

You may also modify the files, but you must then release them as well under the same terms. Credit can be attributed through a link to the creator website: https://www.olimex.com

The software is released under GPL.

It is possible that the pictures in this manual differ from the latest revision of the board.

The product described in this document is subject to continuous development and improvements. All particulars of the product and its use contained in this document are given by OLIMEX in good faith. However all warranties implied or expressed including but not limited to implied warranties of merchantability or fitness for purpose are excluded. This document is intended only to assist the reader in the use of the product. OLIMEX Ltd. shall not be liable for any loss or damage arising from the use of any information in this document or any error or omission in such information or any incorrect use of the product.

This evaluation board/kit is intended for use for engineering development, demonstration, or evaluation purposes only and is not considered by OLIMEX to be a finished end-product fit for general consumer use. Persons handling the product must have electronics training and observe good engineering practice standards. As such, the goods being provided are not intended to be complete in terms of required design-, marketing-, and/or manufacturing-related protective considerations, including product safety and environmental measures typically found in end products that incorporate such semiconductor components or circuit boards.

Olimex currently deals with a variety of customers for products, and therefore our arrangement with the user is not exclusive. Olimex assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein.

THERE IS NO WARRANTY FOR THE DESIGN MATERIALS AND THE COMPONENTS USED TO CREATE OLIMEXINO-STM32. THEY ARE CONSIDERED SUITABLE ONLY FOR OLIMEXINO-STM32.

For product support, hardware information and error reports mail to: support@olimex.com. Note that we are primarily a hardware company and our software support is limited.

Please consider reading the paragraph below about the warranty of Olimex products.

Warranty and returns:

Our boards have lifetime warranty against manufacturing defects and components.

During development work it is not unlikely that you can burn your programmer or development board. This is normal, we also do development work and we have damaged A LOT of programmers and boards during our daily job so we know how it works. If our board/programmer has worked fine then stopped, please check if you didn't apply over voltage by mistake, or shorted something in your target board where the programmer was connected etc. Sometimes boards might get damaged by ESD shock voltage or if you spill coffee on them during your work when they are powered.

Please note that warranty do not cover problems caused by improper use, shorts, over-voltages, ESD shock etc.

If the board has warranty label it should be not broken. Broken labels void the warranty, same applies for boards modified by the customer, for instance soldering additional components or removing components – such boards will be not be a subject of our warranty.

If you are positive that the problem is due to manufacturing defect or component you can return the board back to us for inspection.

When we receive the board we will check and if the problem is caused due to our fault and we will repair/replace the faulty hardware free of charge, otherwise we can quote price of the repair.

Note that all shipping back and forth have to be covered by the customer. Before you ship anything back you need to ask for RMA. When you ship back please attach to it your shipping address, phone, e-mail, RMA# and brief description of the problem. All boards should be sent back in antistatic package and well packed to prevent damages during the transport.