

Mental Noise

Synapse

CV to MIDI Eurorack module

Documentation / Build Guide

SPECIFICATIONS	4
CONTENT OF THE KIT AND FULLY ASSEMBLED MODULE	5
BUILD GUIDE	6
1. Connect the two PCBs together	7
1.1 Insert the male headers into the female headers	8
1.2 Insert the male pins in the middle PCB	8
1.3 Insert the female pins in the back PCB	9
1.4 Solder a few pins on each PCB and make sure both PCBs are aligned	10
1.5 When both PCBs are aligned, solder all the pins	11
2. Solder the jacks	12
2.1 Insert all the jacks on the PCB	13
2.2 Install the faceplate on the jacks and screw the nuts	14
2.3 Solder the jacks to the PCB	15
3.0 Install the 5V regulator	16
3.1 Solder the resistors	17
3.2 Solder the capacitors	18
3.3 Solder the L7805 5V regulator	19
4. Install the connectors	20
4.1 Solder the 2x3 pin IDC connector (extension port)	21
4.2 Solder the 2x5 pin IDC connector (Eurorack connector)	21
4.3 Solder the 1x3 pin female header (will be connected to the MIDI connector)	22
4.4 Solder the Arduino headers (2x 1x15 pin female headers)	23
5. Solder extra resistors on PCB revision 1.1	24
5.1 Solder all the resistors on the right pins	25
5.2 Solder the first resistor to 4 th pin from the top in the other column of pins	26
5.3 Solder the next resistor to the first one	26
5.4 Do the same for the other resistors	27
6. Prepare the MIDI connector	28
5.1 Solder the wires to the MIDI connector	29
5.2 Heat shrink what you just soldered	30
5.3 Solder the 1x3 pin male connector to the other end of the wires	30
7. Install the MIDI connector	31
7.1 Install the MIDI connector in the faceplate	32
7.2 Connect the MIDI connector to the PCB with the Arduino	33
7.3 Install the Arduino on top of the wires (or not, it just looks neater to me)	34

FIRMWARE UPLOAD	35
1. Download the Arduino IDE	35
2. Download the firmware on Github	36
3. Open the firmware	37
4. Configure the Arduino IDE to use the right Arduino board	39
4.1 Choose the right board	39
4.2 Choose the right processor	40
4.3 Choose the right port	41
5. Add the required libraries in the Arduino IDE	42
6. Upload the firmware	44
TIME TO PATCH!	46

Specifications

This module is designed to be used in an Eurorack system, it requires a +12V and a ground connection to the power supply.

The dimensions are:

- Width: 6HP
- Height: 3U
- Depth: 40mm

The MIDI device is controlled by this module using a MIDI cable (5-pin DIN connector).

Two types of inputs are available on the module:

- **CV x6**
Analog input used to control parameters of the MIDI device.
The MIDI command sent by those inputs is a Control Change.
Can be any voltage between 0 and +5V.
Higher voltages (up to +10V) will be clipped to +5V.
- **Gate x10**
Digital input used to trigger a note on the MIDI device.
The MIDI command sent by those inputs is a Note ON.
A pulse of 20ms is triggered by any gate or trigger signal over 3V (up to +10V).

The module is controlled by an Arduino Nano (clone) loaded with the firmware available on
github : <https://github.com/atulrnt/mental-noise-synapse>

This firmware can easily be modified to fit your needs and your specific MIDI device.

I might consider building a web app to allow easy customization of the firmware depending
on the module popularity.

Content of the kit and fully assembled module

The kit contains all the parts and components required to build the module.

Here are all the parts included in the kit:

- 1x PCB with pre-soldered SMD components (called “middle PCB” in the documentation)
- 1x Unpopulated PCB (called “back PCB” in the documentation)
- 1x faceplate in PCB material
- 1x Arduino Nano (clone with no pre-loaded firmware)
- 1x USB cable to connect the Arduino Nano to a computer
- 1x 100 Ohm 1/4W resistor
- 2x 220 Ohm 1/4W resistor
- 6x 10k Ohm 1/4W resistor (only for r1.1)
- 1x 100nF ceramic capacitor
- 1x 10µF electrolytic capacitor
- 1x L7805 5V voltage regulator
- 16x Mono jack socket
- 16x Knurled nut
- 1x MIDI 5-pin DIN connector
- 3x wires to connect the MIDI connector to the module
- 6x heat shrink pieces
- 2x hex screws and nut to screw the MIDI connector on the module
- 2x 1x15 pin female header
- 3x 2x2 pin female header
- 1x 1x3 pin female header
- 1x 2x3 pin female header
- 6x 1x2 pin male header
- 3x 1x3 pin male header
- 1x 2x5 pin IDC connector (10 pin Eurorack connector)
- 1x 2x3 pin IDC connector (6 pin extension connector)
- 1x 10 pin to 16 pin or 10 pin to 10 pin Eurorack cable depending on your needs

The fully assembled module includes:

- 1x Fully assembled and tested module with pre-loaded firmware (I can even customize the firmware for you to a certain extent if you ask nicely)
- 1x 10 pin to 16 pin or 10 pin to 10 pin Eurorack cable depending on your needs

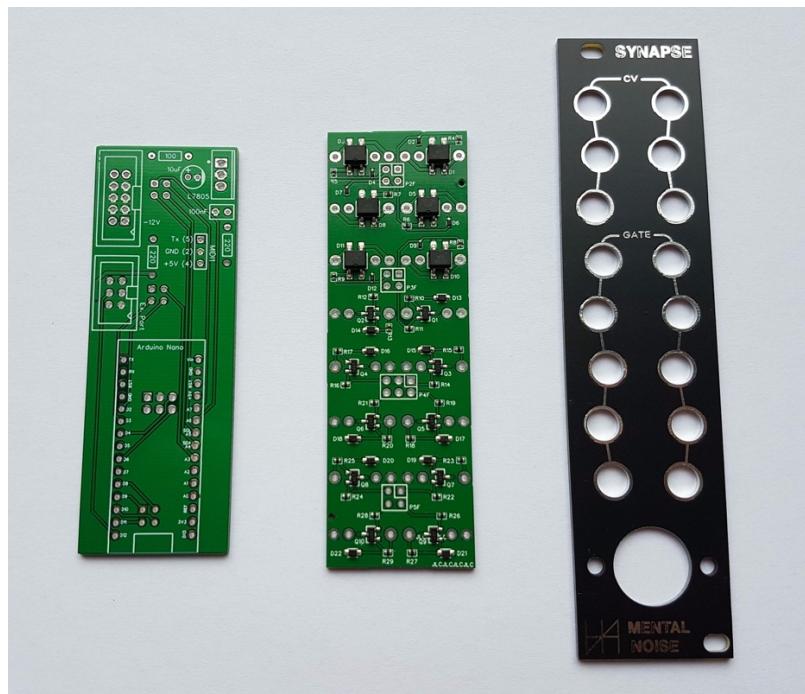
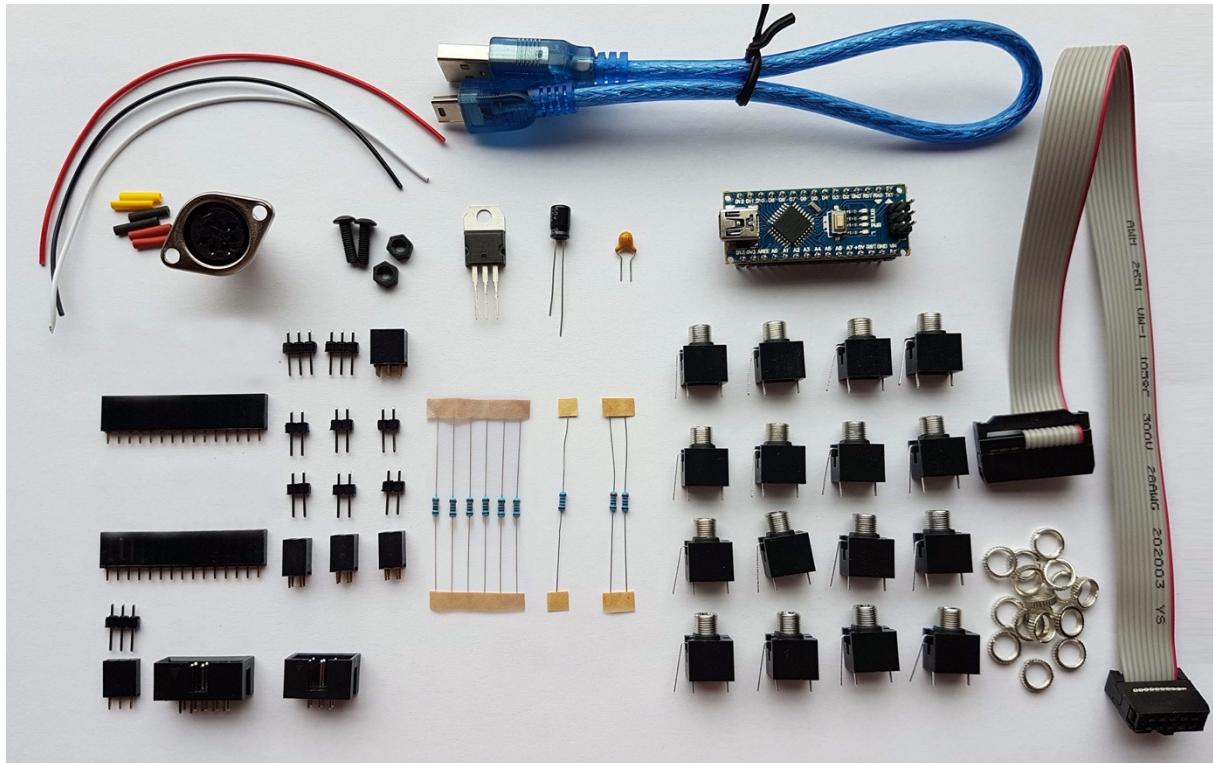
Parts you will need that are not included in the kit or the fully assembled module:

- 1x MIDI 5-pin DIN cable
- 2x M3 or M2.5 screws depending on your rack

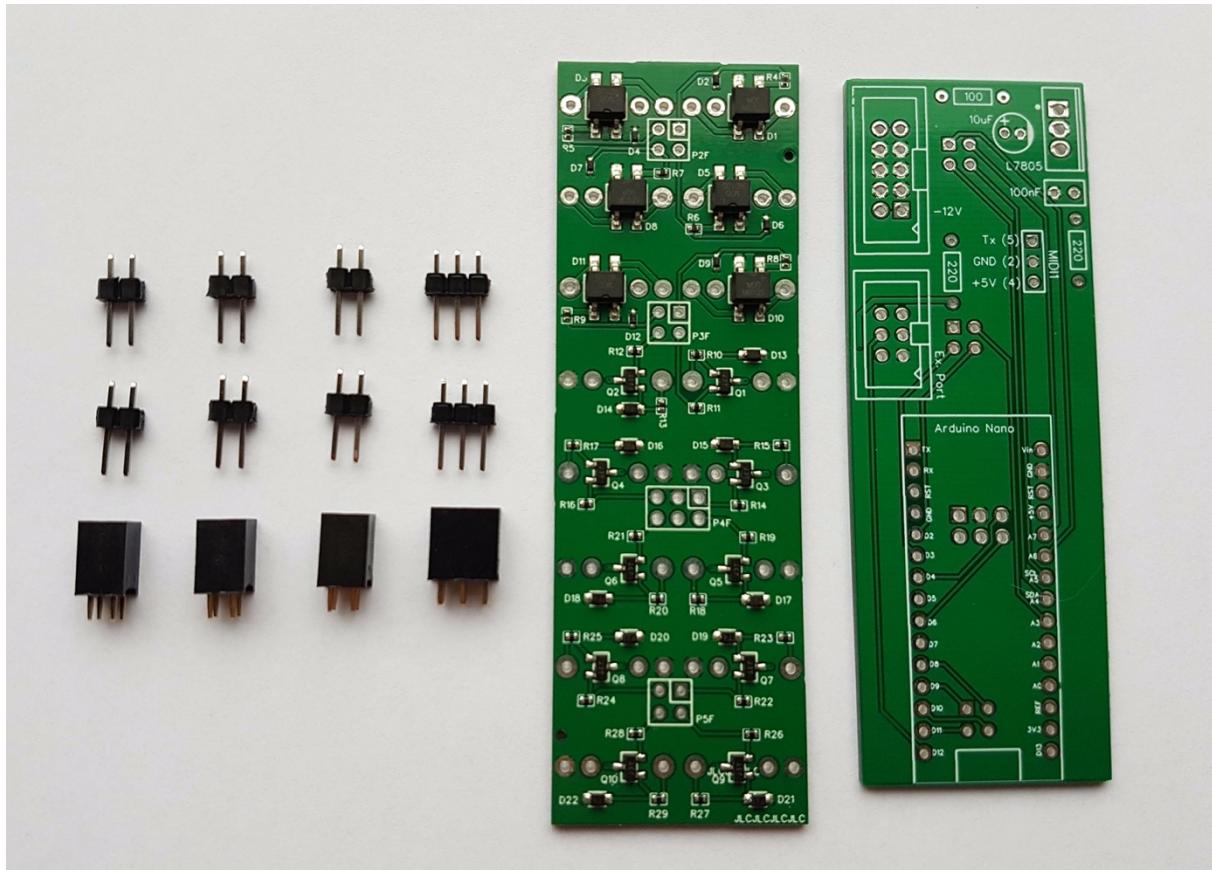
Build guide

With the kit you will receive all the parts listed previously.

This module is really easy to build even for a beginner if you take your time and follow this guide carefully.



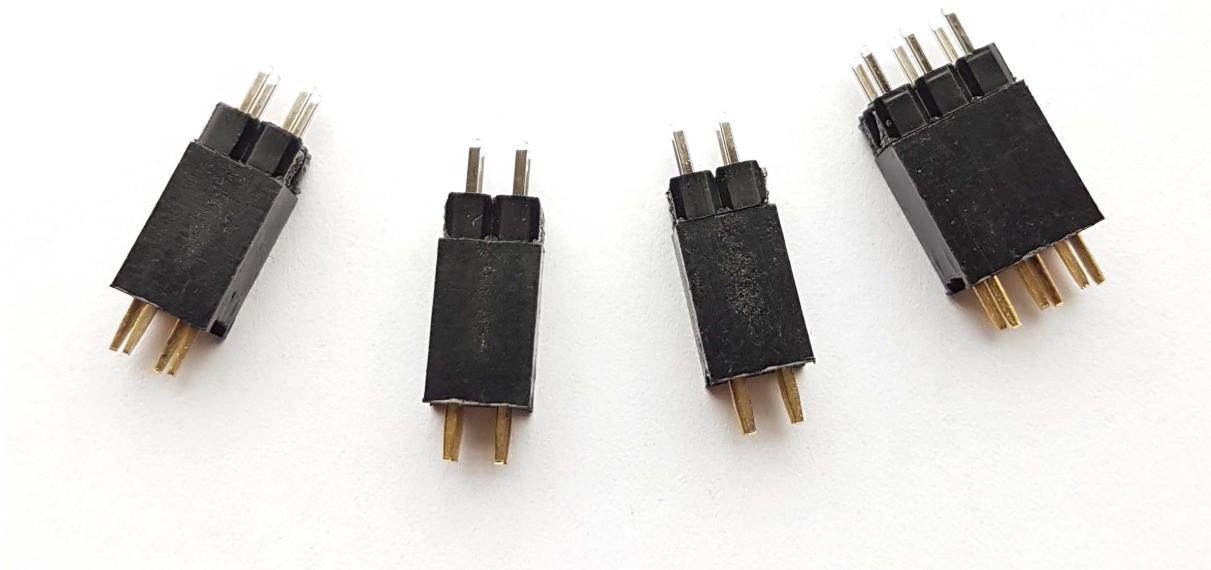
1. Connect the two PCBs together



For this step you will need:

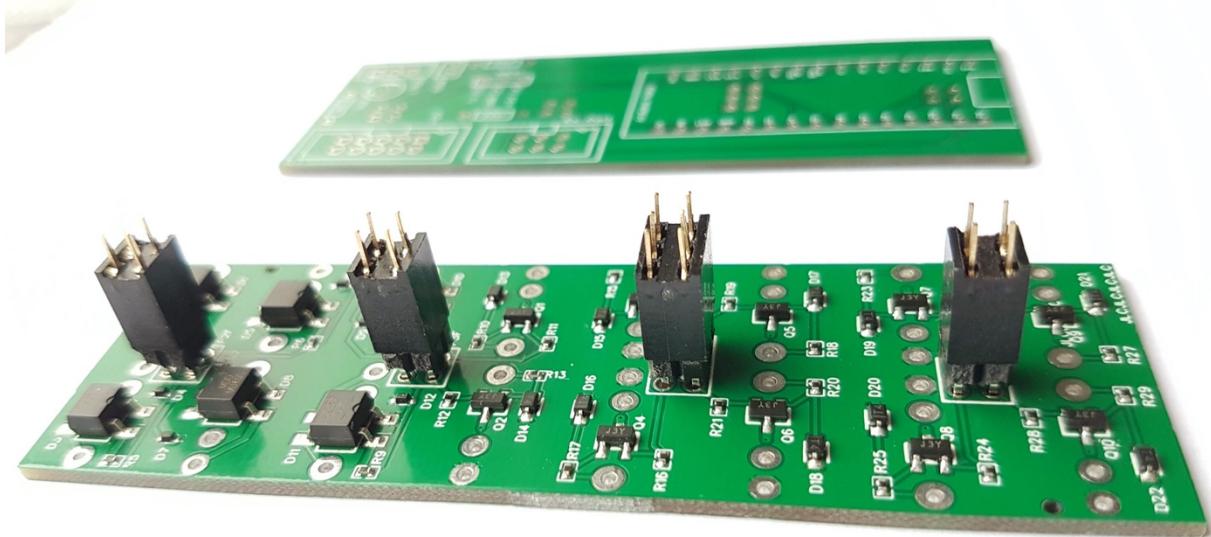
- The middle PCB
- The back PCB
- 3x 2x2 pin female header
- 1x 2x3 pin female header
- 6x 1x2 pin male header
- 2x 1x3 pin male header

1.1 Insert the male headers into the female headers

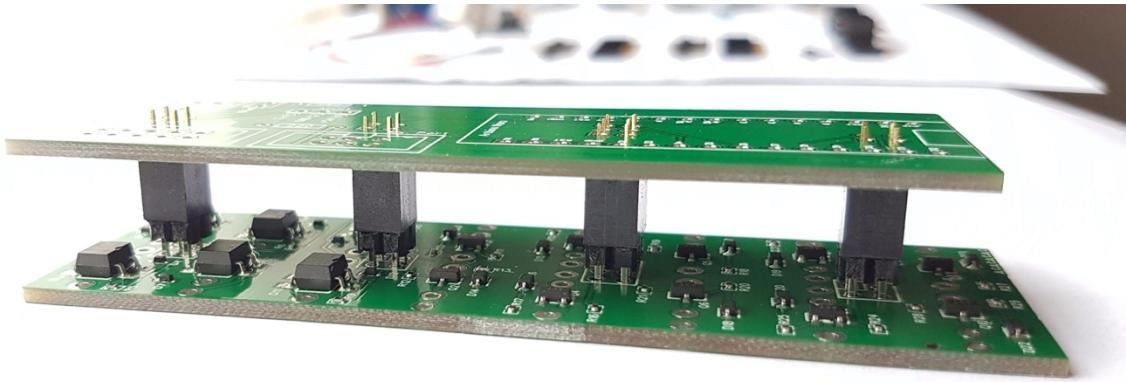


1.2 Insert the male pins in the middle PCB

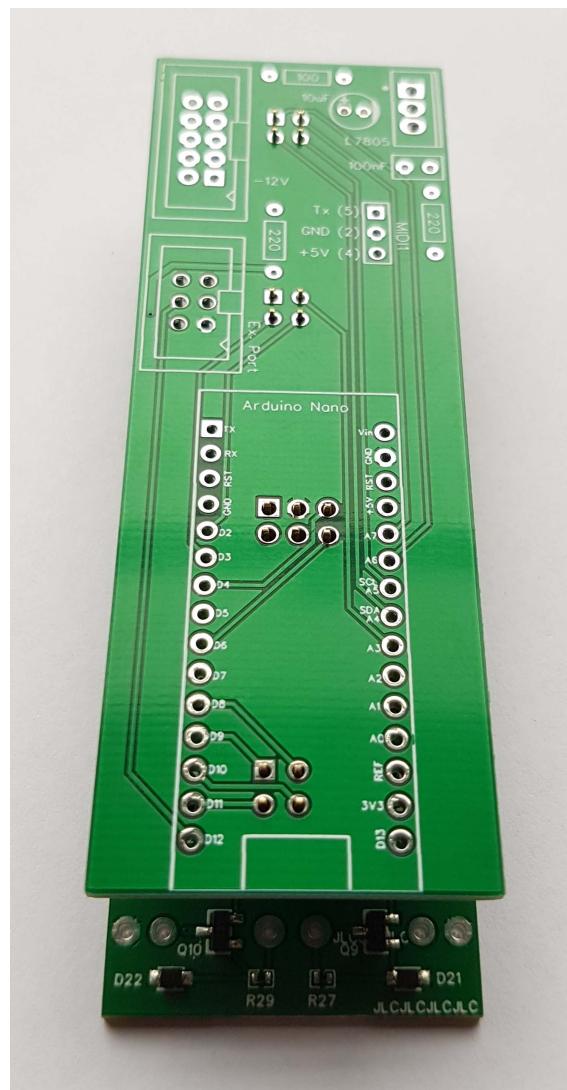
Do not solder anything yet



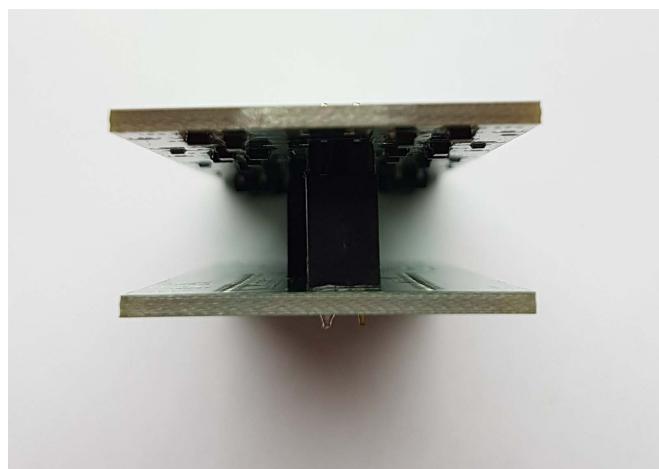
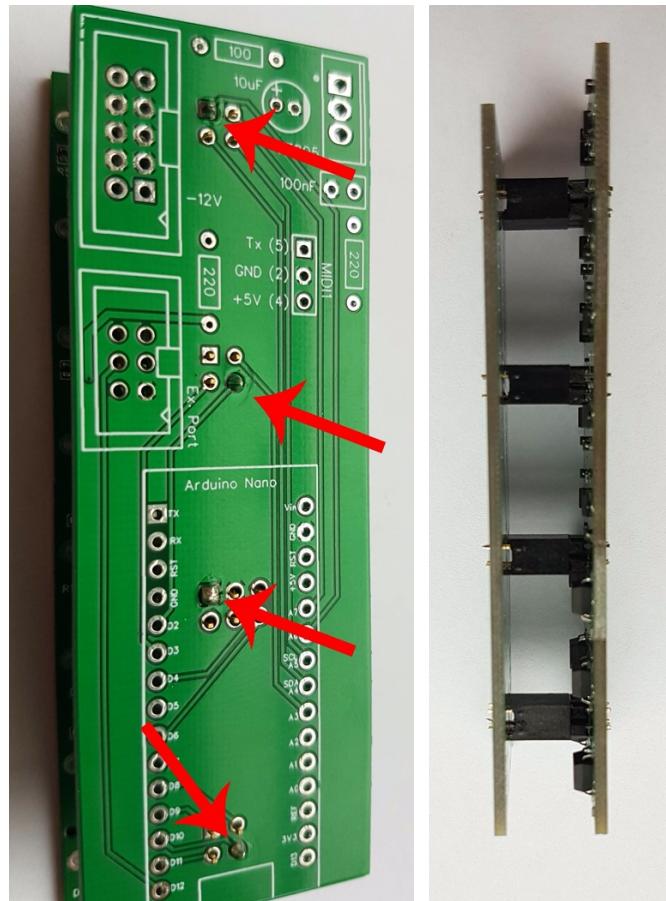
1.3 Insert the female pins in the back PCB



The Arduino image should still be visible to you, if it's not, flip the upper.



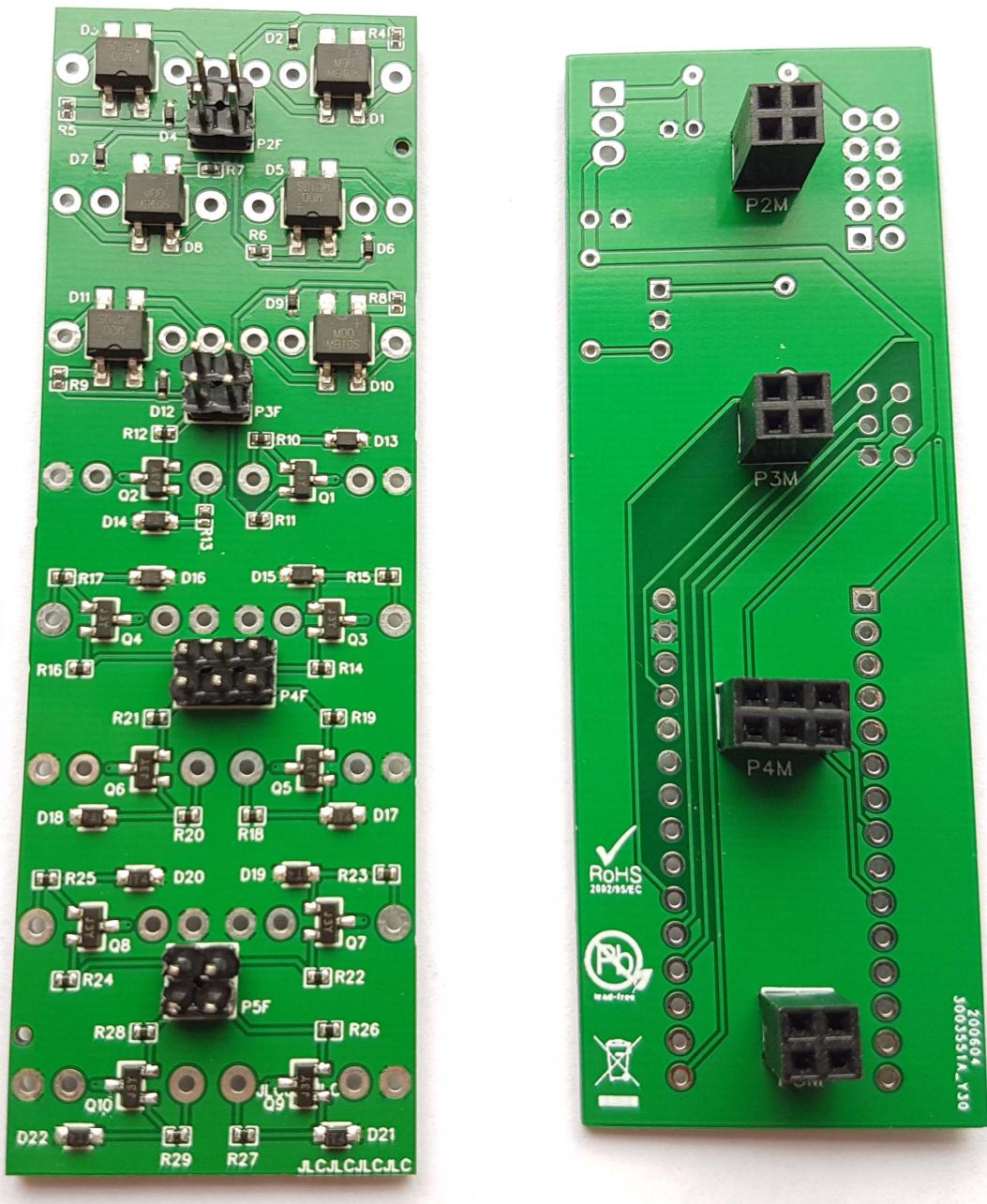
1.4 Solder a few pins on each PCB and make sure both PCBs are aligned



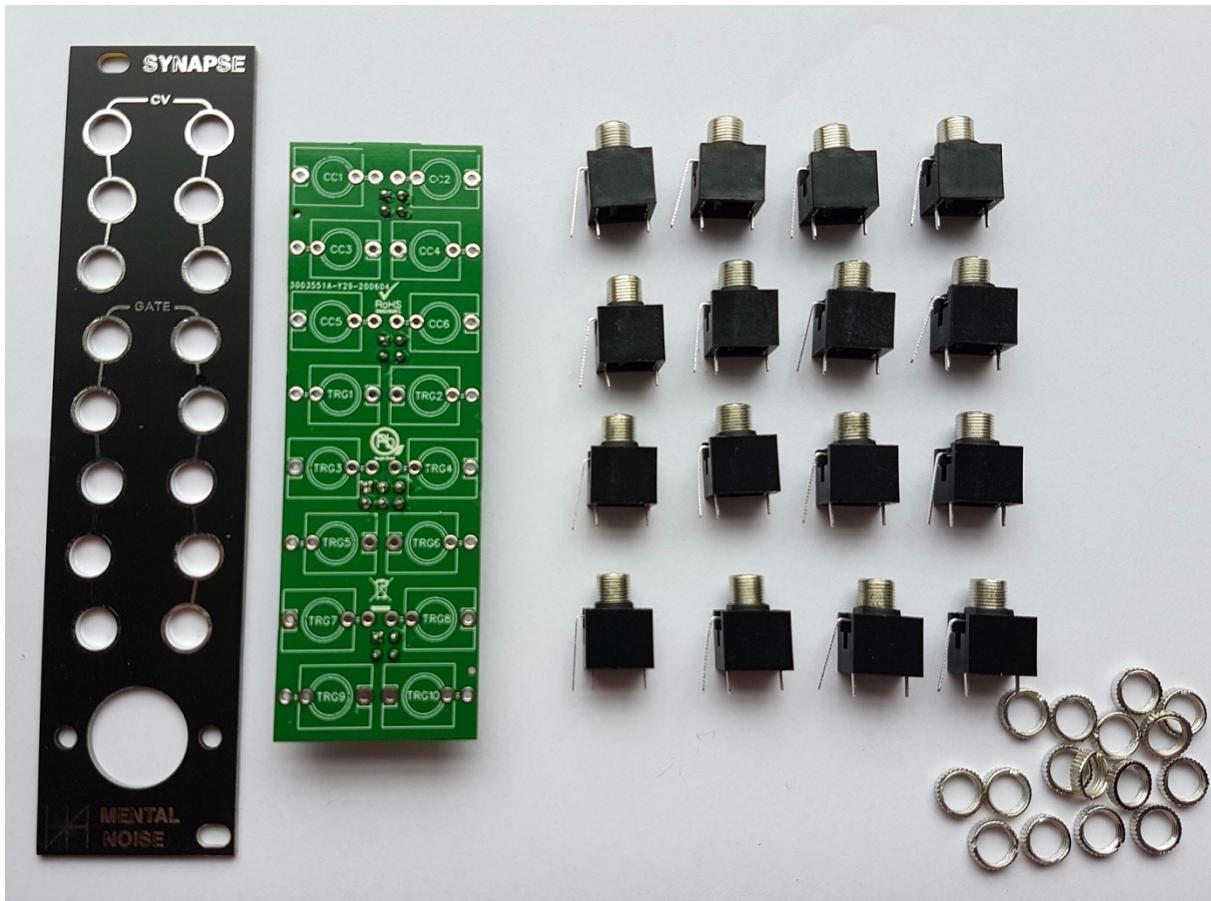
1.5 When both PCBs are aligned, solder all the pins

When soldering those headers, be careful not to burn or unsolder the small SMD components already in place!

Once you soldered all the headers, it doesn't matter much if they are perfectly aligned with their symbol on the PCBs or not, the important part is that the female headers are aligned with the male headers.



2. Solder the jacks

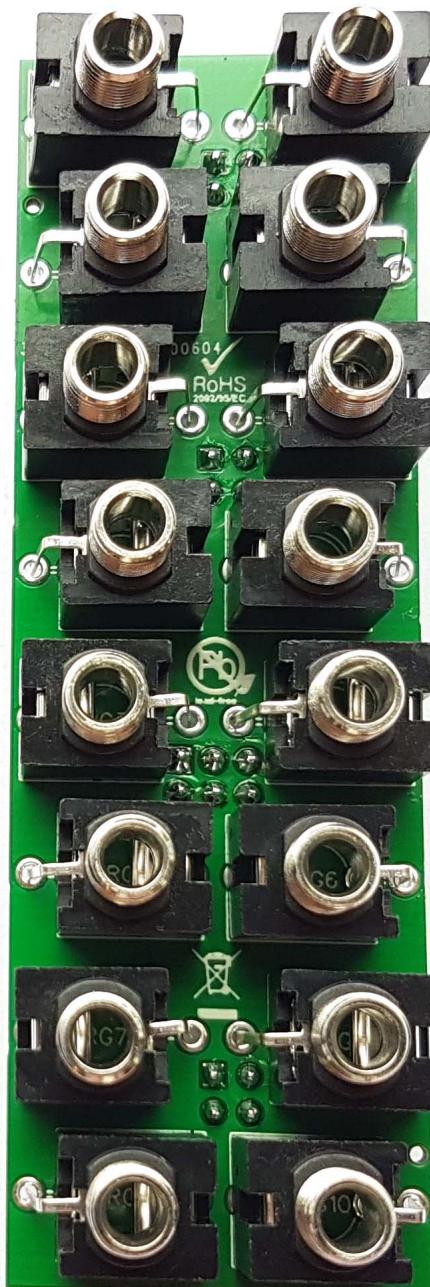


For this step you will need:

- The middle PCB
- The faceplate
- 16x Mono jack socket
- 16x Knurled nut

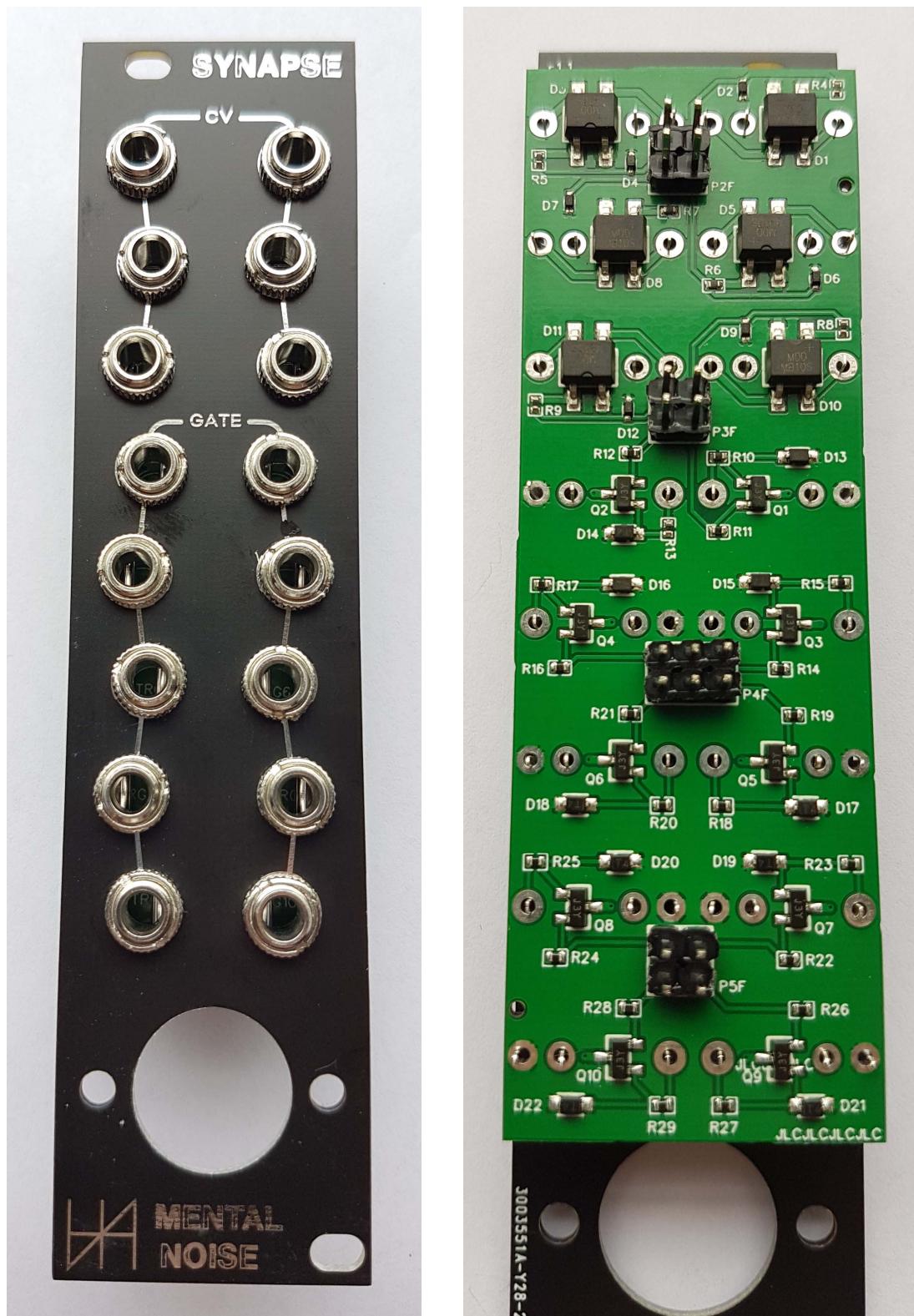
2.1 Insert all the jacks on the PCB

Do not solder anything yet



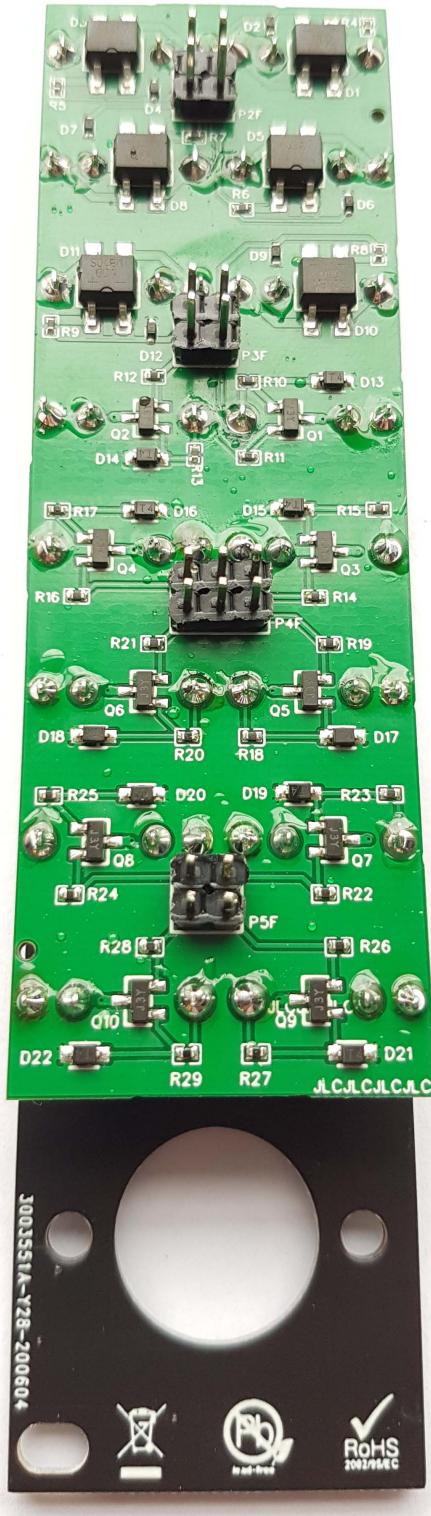
2.2 Install the faceplate on the jacks and screw the nuts

The faceplate, the jacks and the PCB should be aligned correctly at this point.

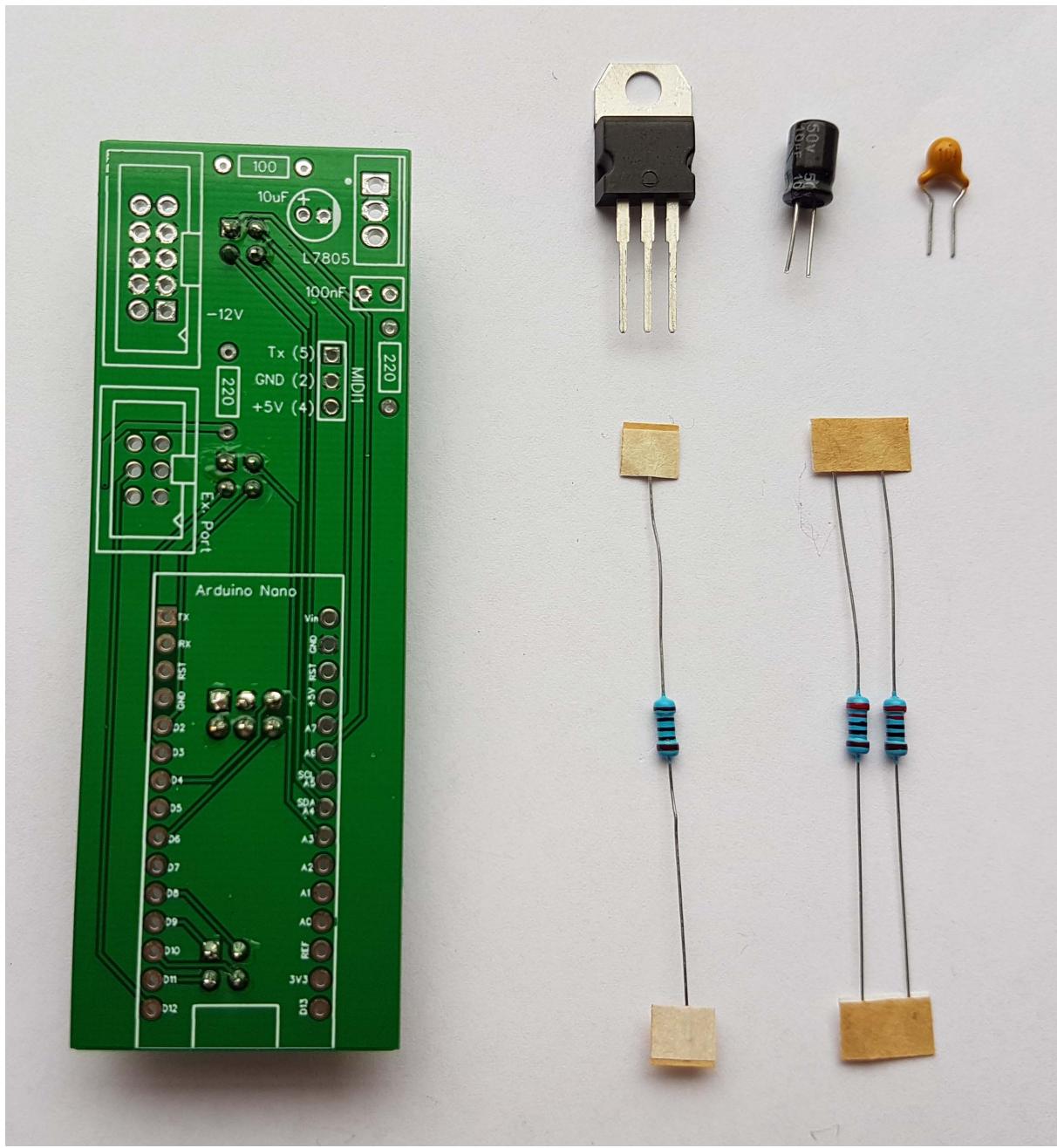


2.3 Solder the jacks to the PCB

When soldering the jacks, be careful not to burn or unsolder the small SMD components already in place!



3.0 Install the 5V regulator

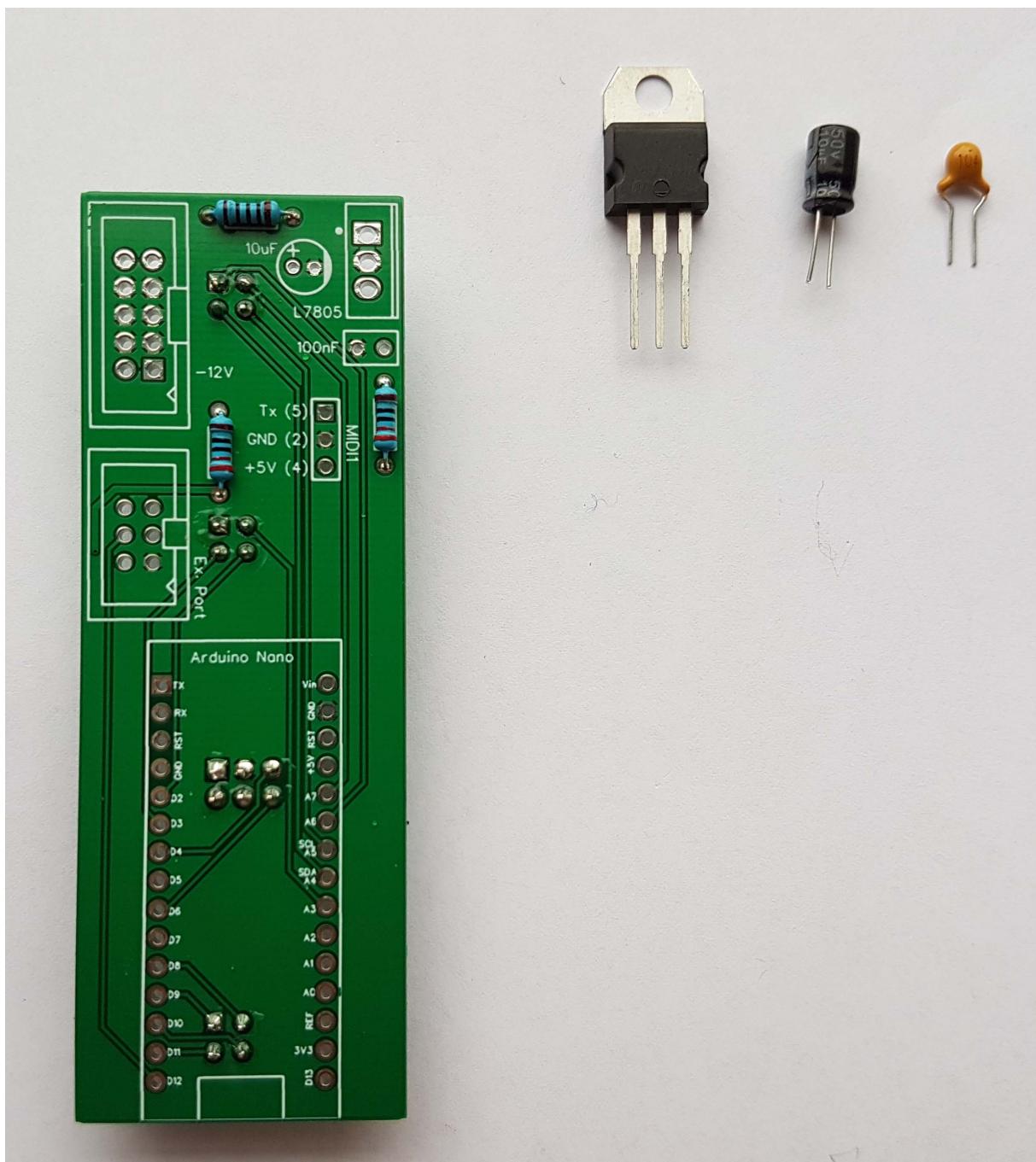


For this step you will need:

- The back PCB
- 1x 100 Ohm 1/4W resistor
- 2x 220 Ohm 1/4W resistor
- 1x 100nF ceramic capacitor
- 1x 10 μ F electrolytic capacitor
- 1x L7805 5V voltage regulator

3.1 Solder the resistors

The resistor values are indicated on the PCB but if you have a doubt, the top one is 100 Ohm, the two others are 220 Ohm.

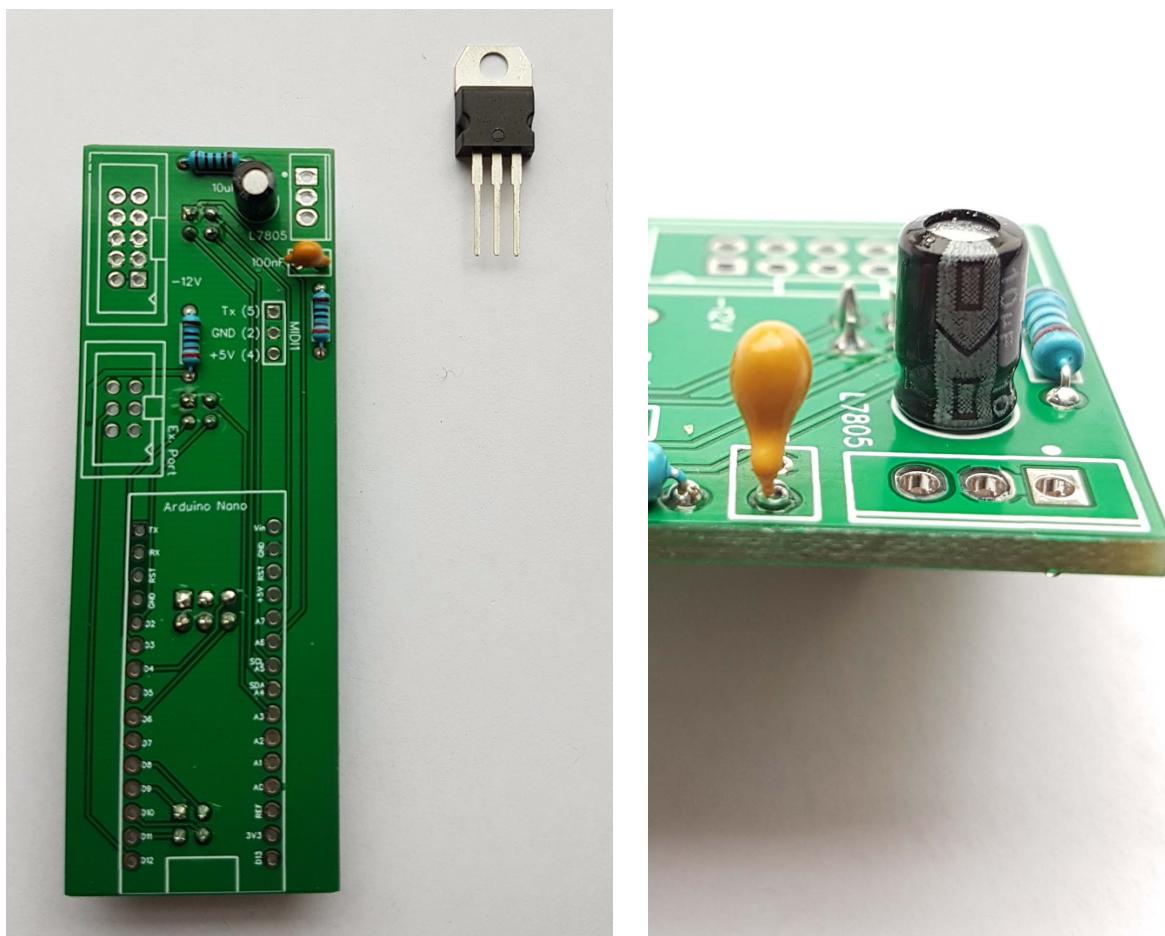


3.2 Solder the capacitors

The 100nF ceramic capacitor (the small yellow one) can be installed in any direction as it doesn't have polarity.

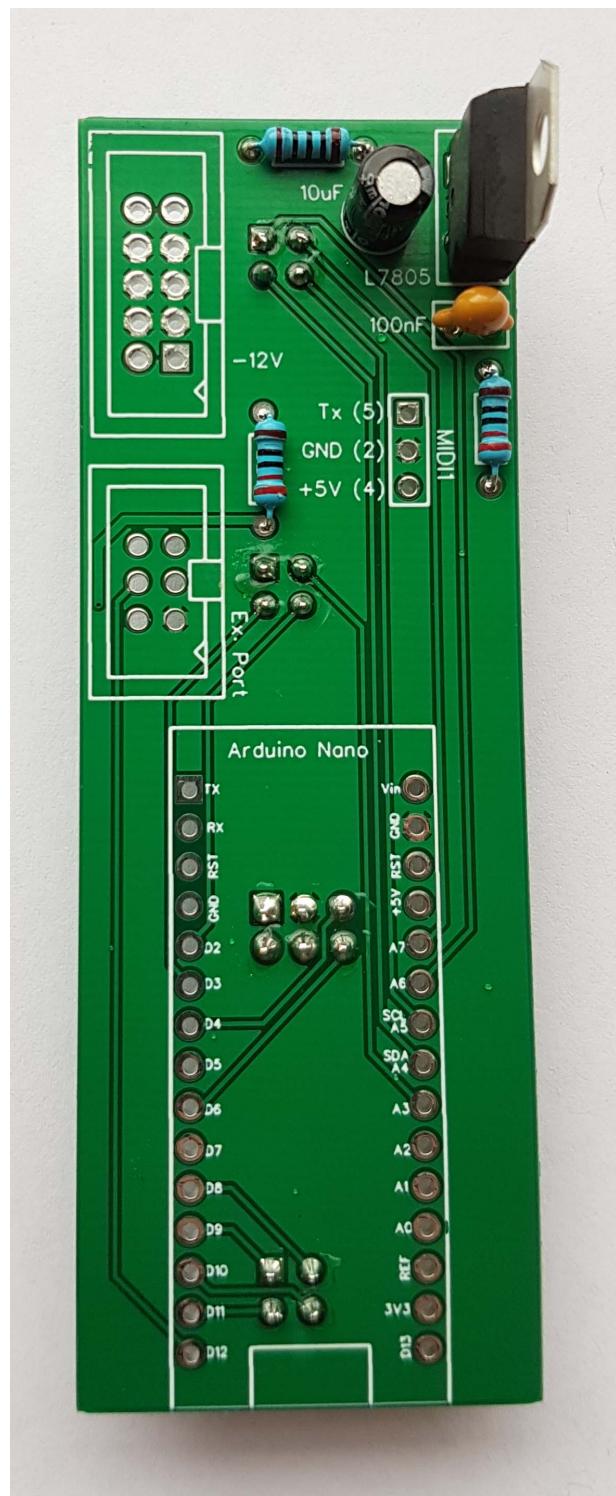
However, **be careful with the 10 μ F electrolytic capacitor** (the bigger black one), **it is important to respect the polarity for this one** or it will blow up when the module will be powered-up.

The white stripe on the capacitor needs to match the white stripe on the PCB.

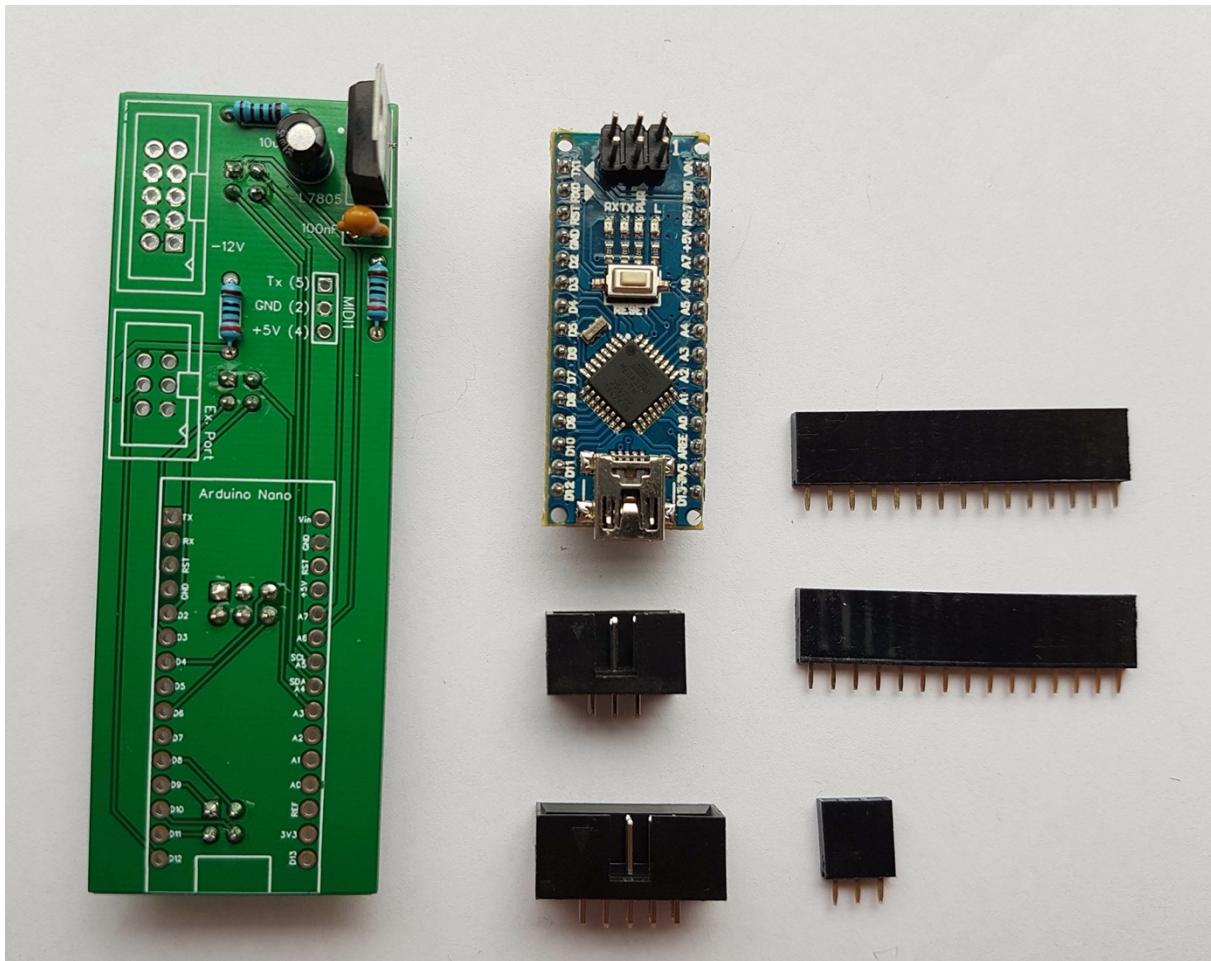


3.3 Solder the L7805 5V regulator

The heatsink (metal part) of the L7805 needs to face the outside of the PCB.



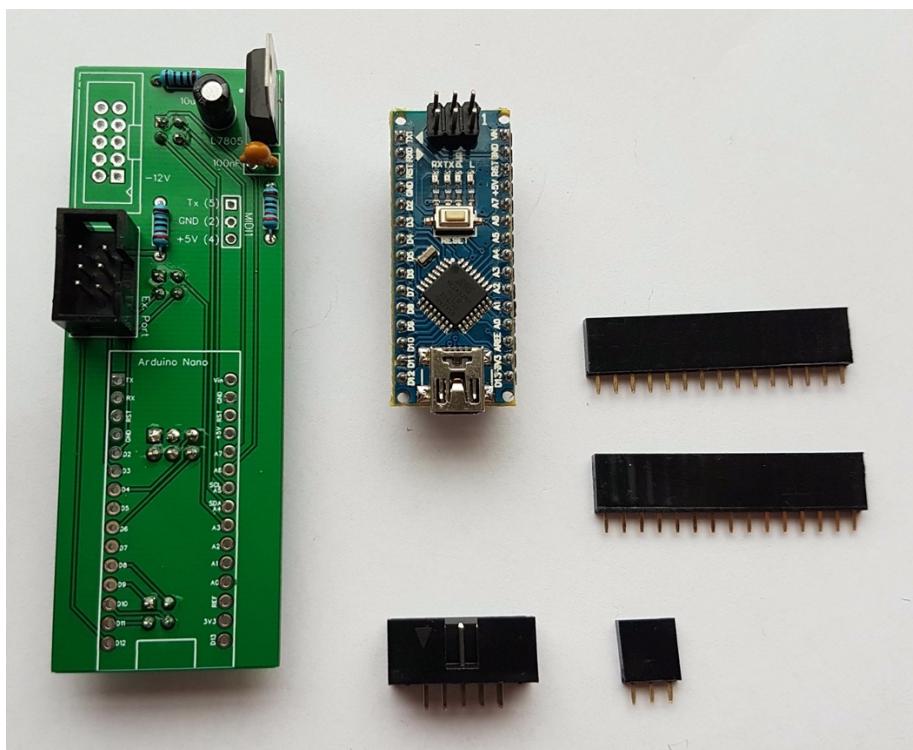
4. Install the connectors



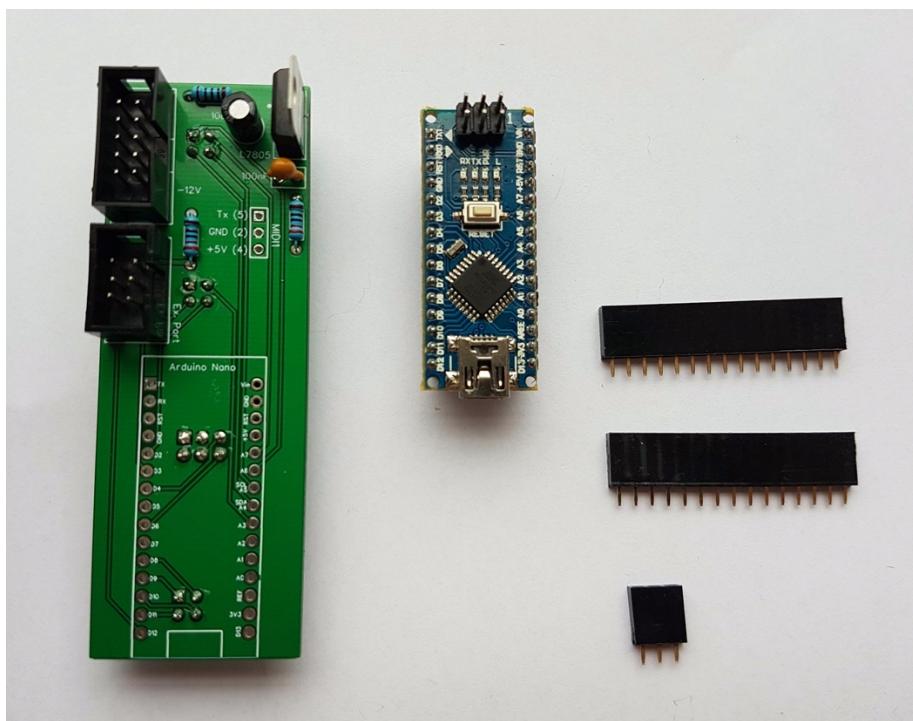
For this step you will need:

- The back PCB
- The Arduino Nano
- 2x 1x15 pin female header
- 1x 1x3 pin female header
- 1x 2x5 pin IDC connector (10 pin Eurorack connector)
- 1x 2x3 pin IDC connector (6 pin extension connector)

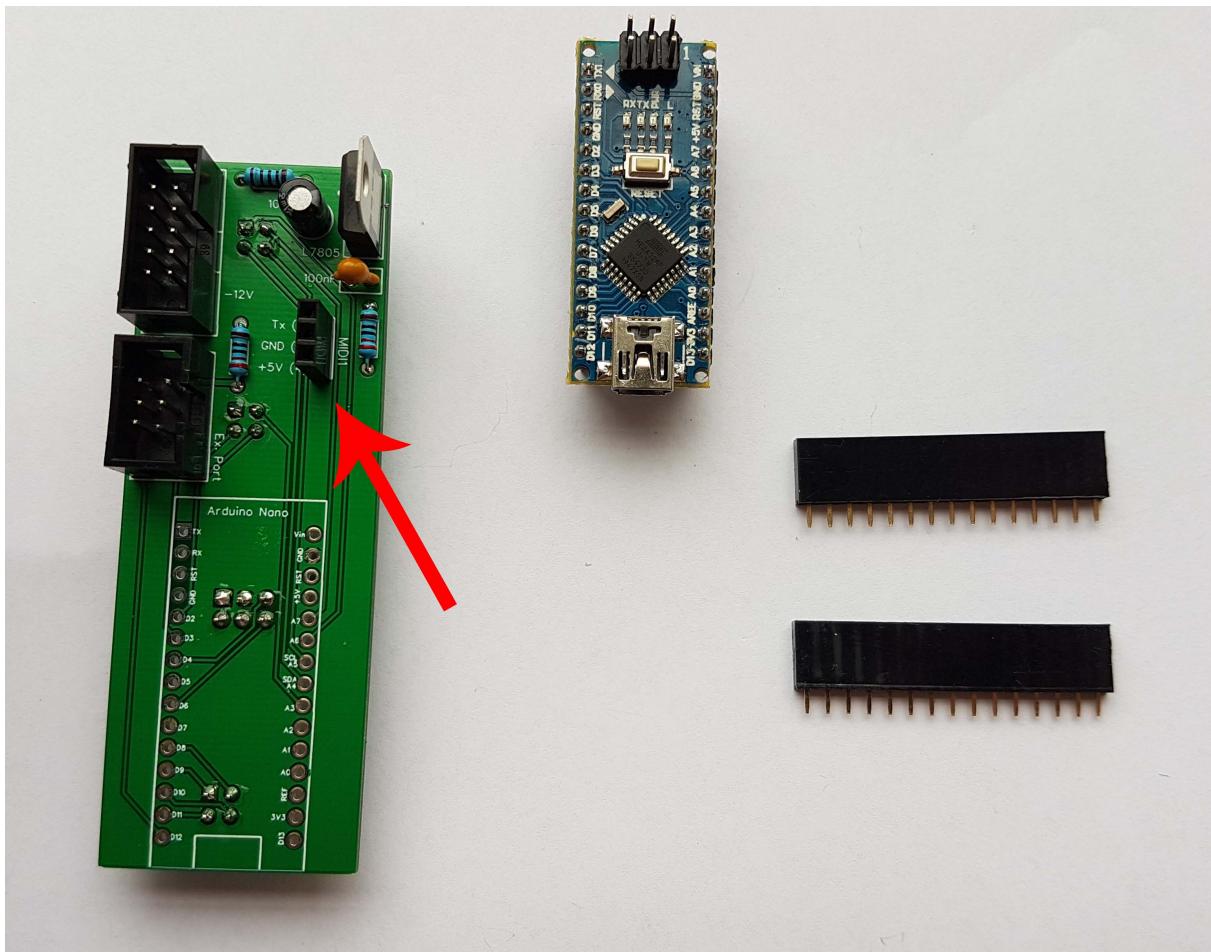
4.1 Solder the 2x3 pin IDC connector (extension port)



4.2 Solder the 2x5 pin IDC connector (Eurorack connector)



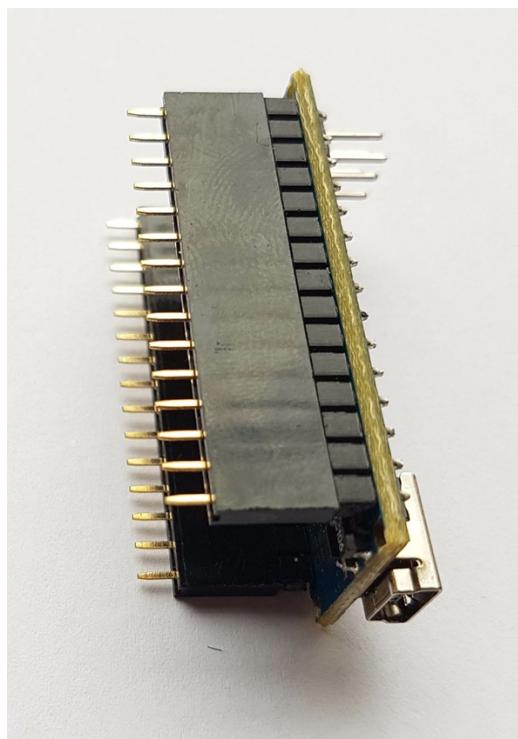
4.3 Solder the 1x3 pin female header (will be connected to the MIDI connector)



4.4 Solder the Arduino headers (2x 1x15 pin female headers)

First, install the female headers on the Arduino and then install the Arduino on the PCB, this way you are sure everything is correctly aligned and will fit again later.

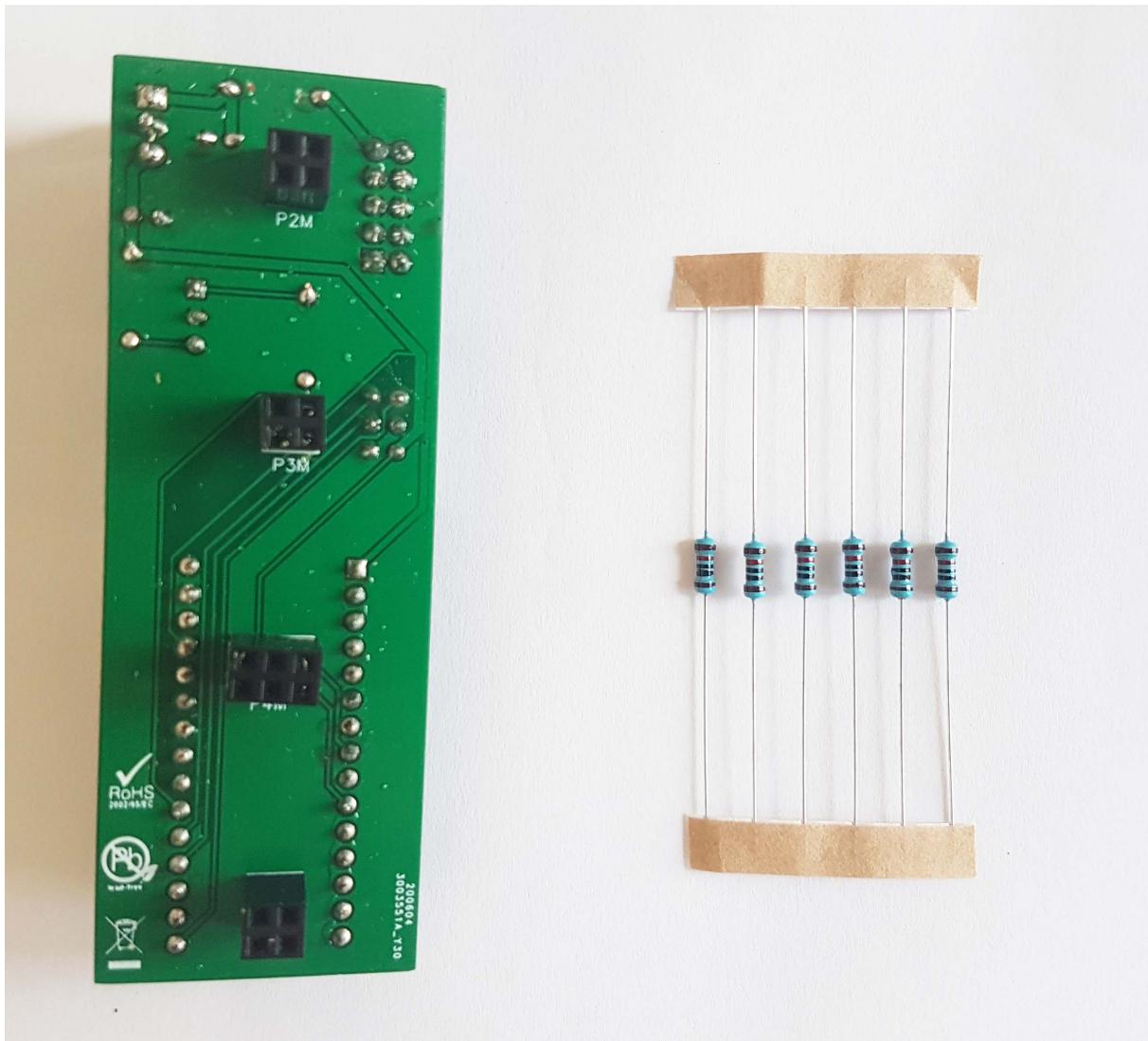
When you are happy with the alignment you can solder the headers to the board with the Arduino still installed in the headers during the soldering.



5. Solder extra resistors on PCB revision 1.1

If your PCB is of revision 1.1 (no revision marking on the middle PCB), you will need to solder extra resistors that were forgotten during the design of the board.

Once soldered, the module in revision 1.1 will work the same as a module in revision 1.2.



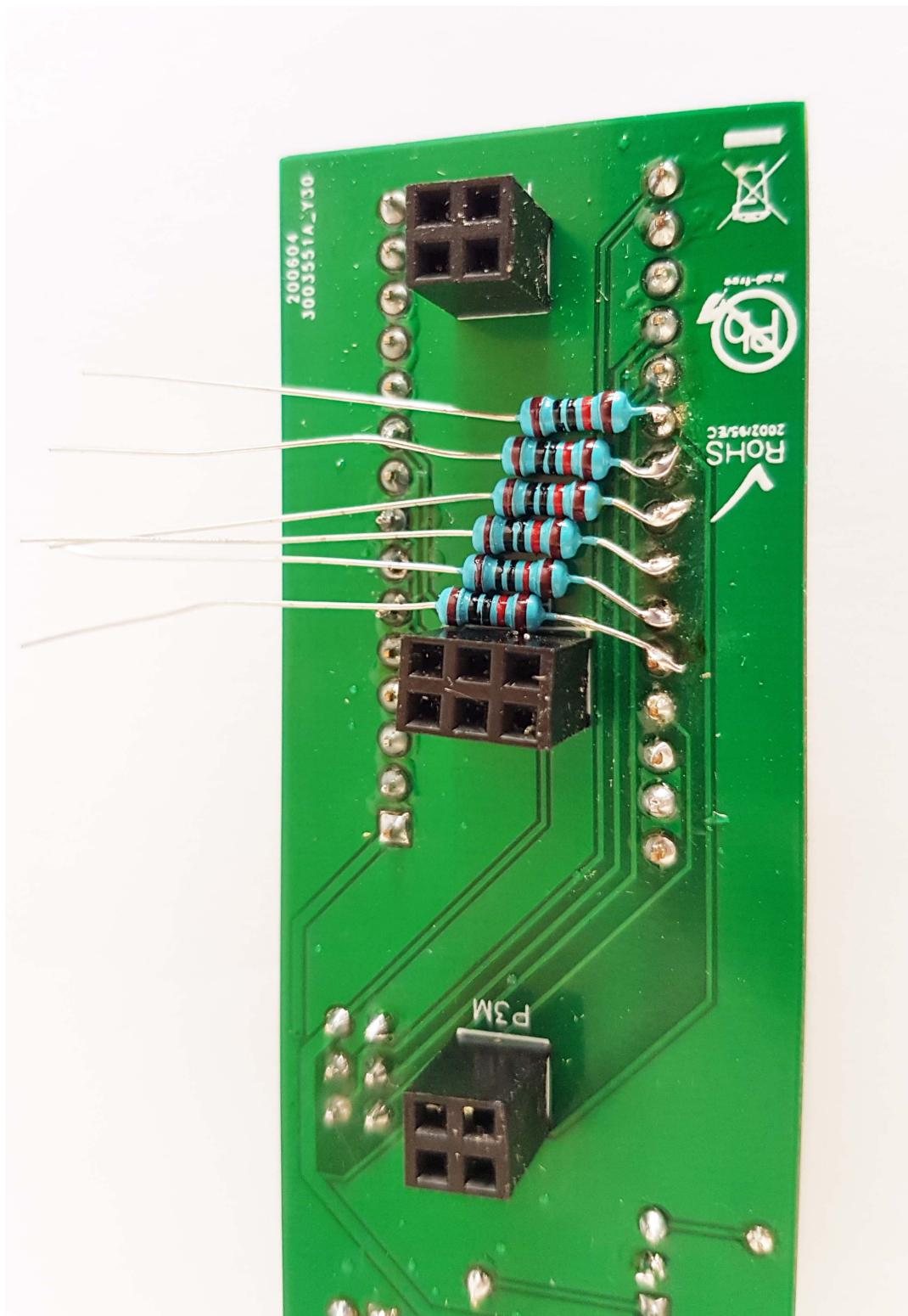
For this step you will need:

- The middle PCB
- 6x 10k Ohm resistor 1/4W

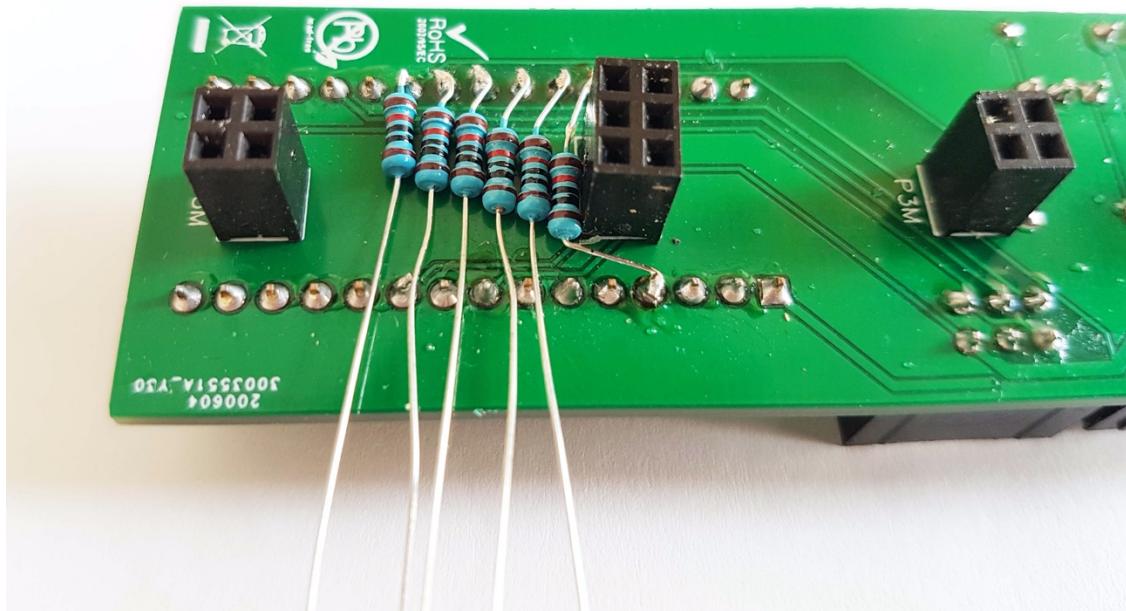
5.1 Solder all the resistors on the right pins

Solder the resistors one by one to each pin as on this picture.

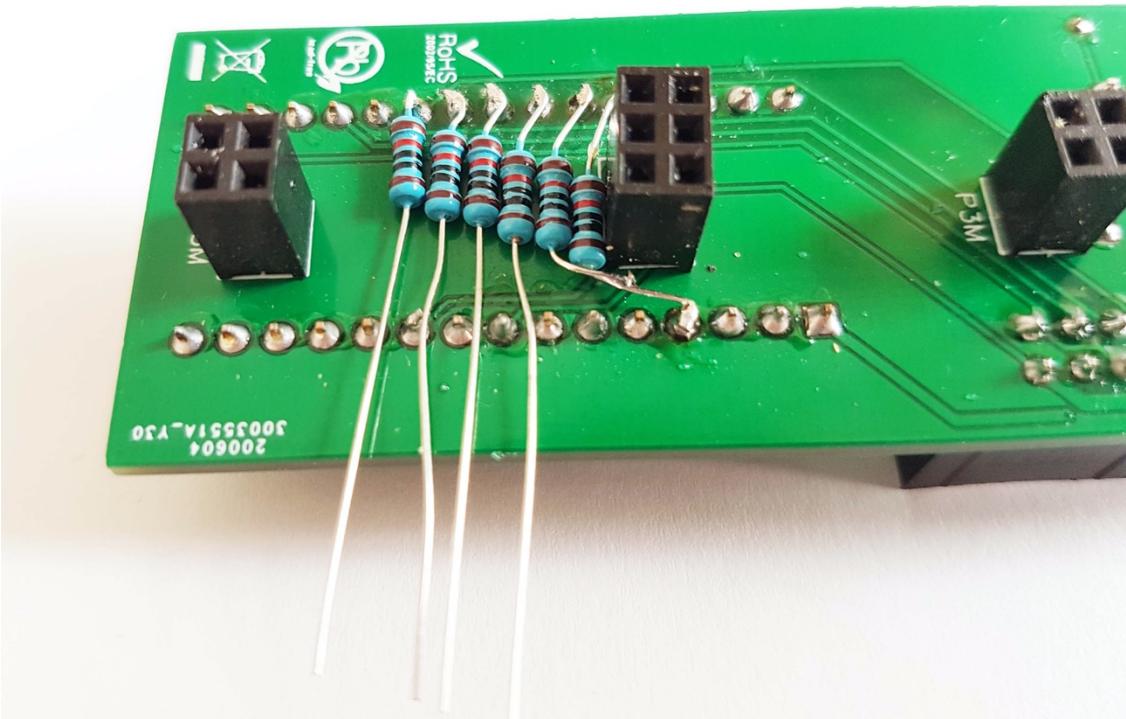
On the picture the PCB is upside down, you start at the 6th pin from the bottom of the board.



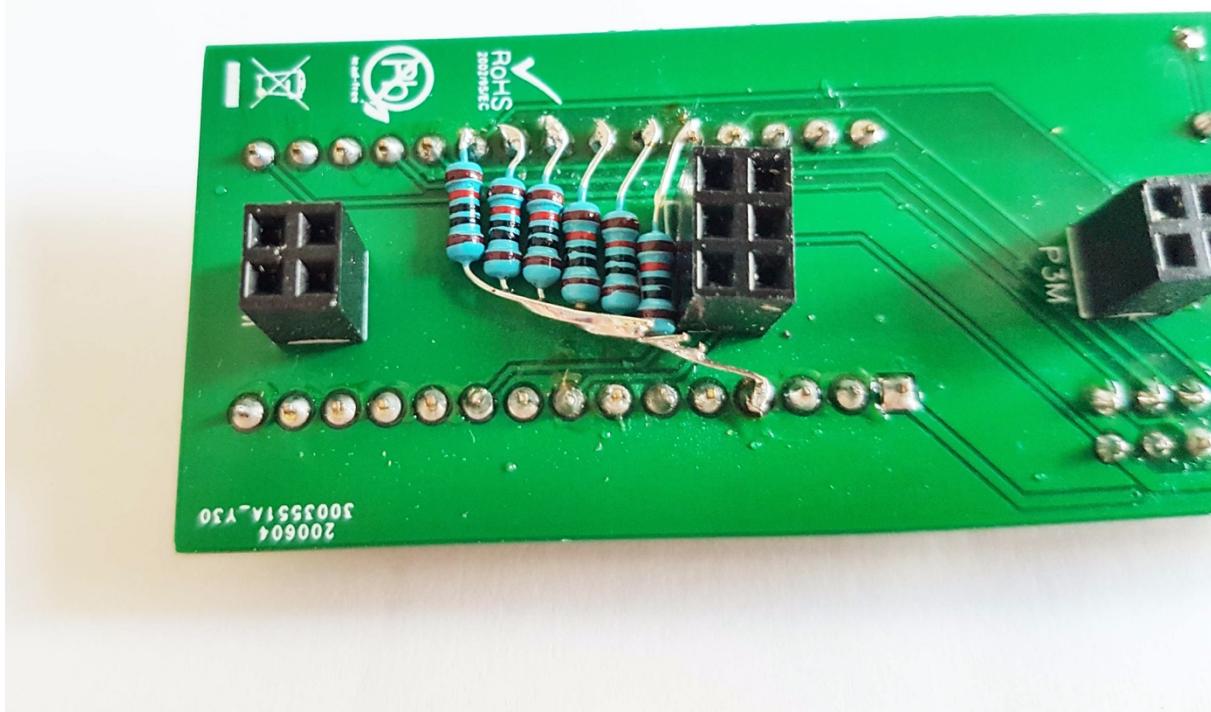
5.2 Solder the first resistor to 4th pin from the top in the other column of pins



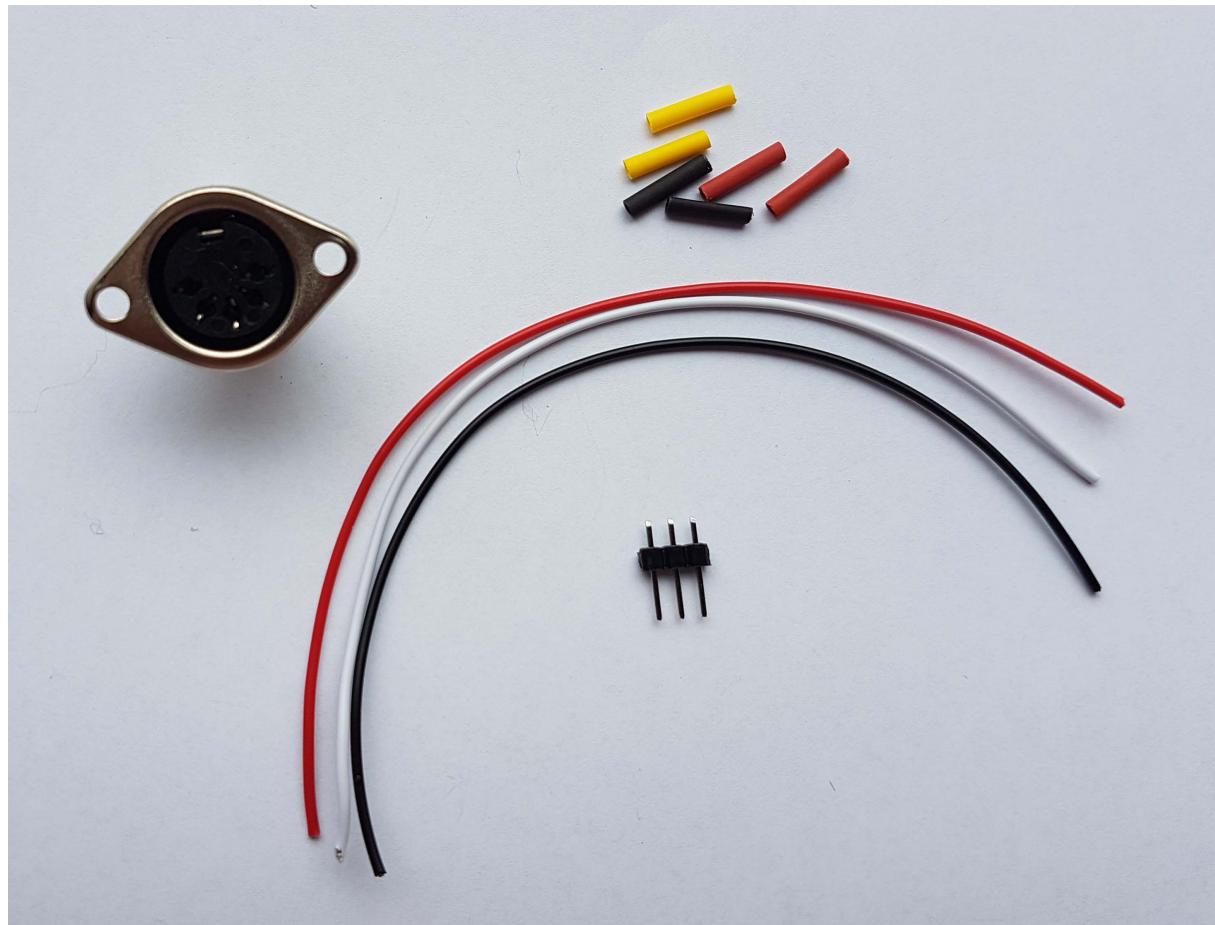
5.3 Solder the next resistor to the first one



5.4 Do the same for the other resistors



6. Prepare the MIDI connector



For this step you will need:

- 1x MIDI 5-pin DIN connector
- 3x wires to connect the MIDI connector to the module
- 6x heat shrink pieces
- 1x 1x3 pin male header

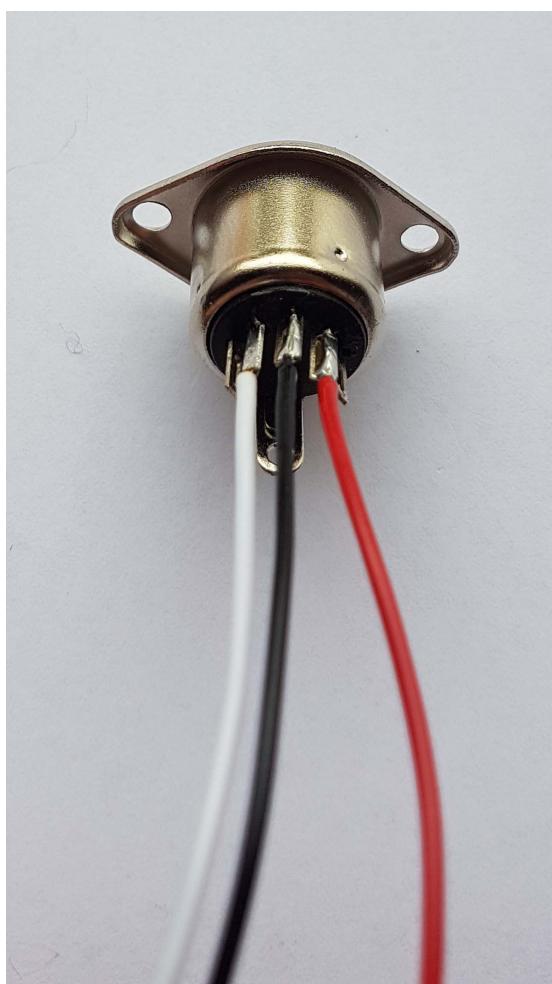
5.1 Solder the wires to the MIDI connector

Pre-tin the 3 wires to make it easier to solder them on the MIDI connector pads.

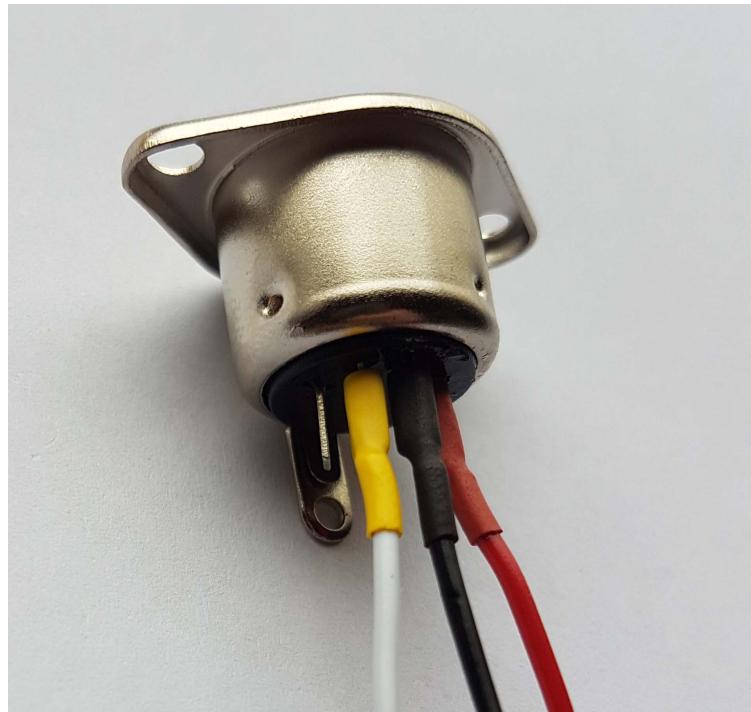
The red wire (5V) goes to MIDI pin 4.

The white wire (Tx) goes to MIDI pin 5.

The black wire (GND) goes to MIDI pin 2.

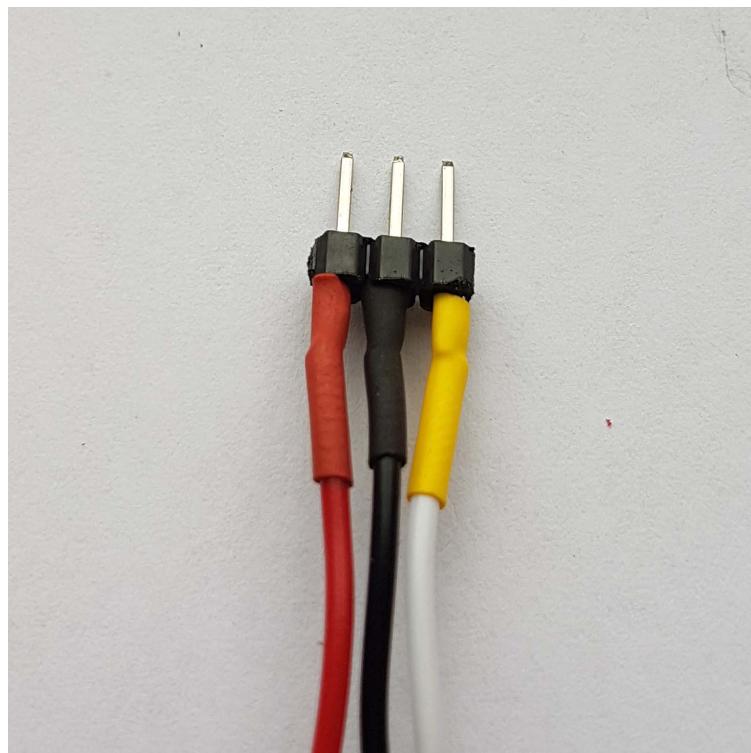


5.2 Heat shrink what you just soldered

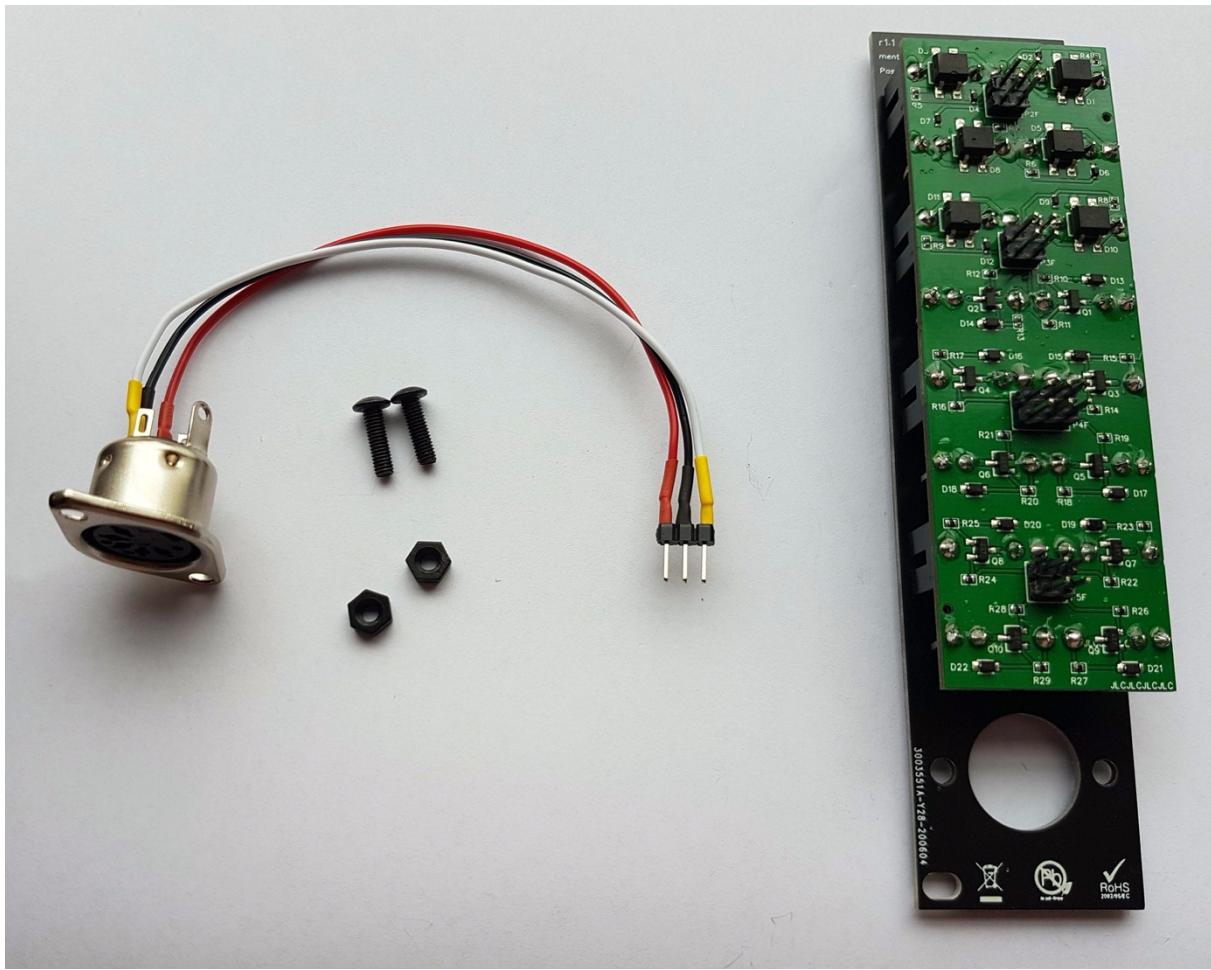


5.3 Solder the 1x3 pin male connector to the other end of the wires

Insert the 3 other heat shrink pieces on the wires before soldering the 3 pins header ;)



7. Install the MIDI connector

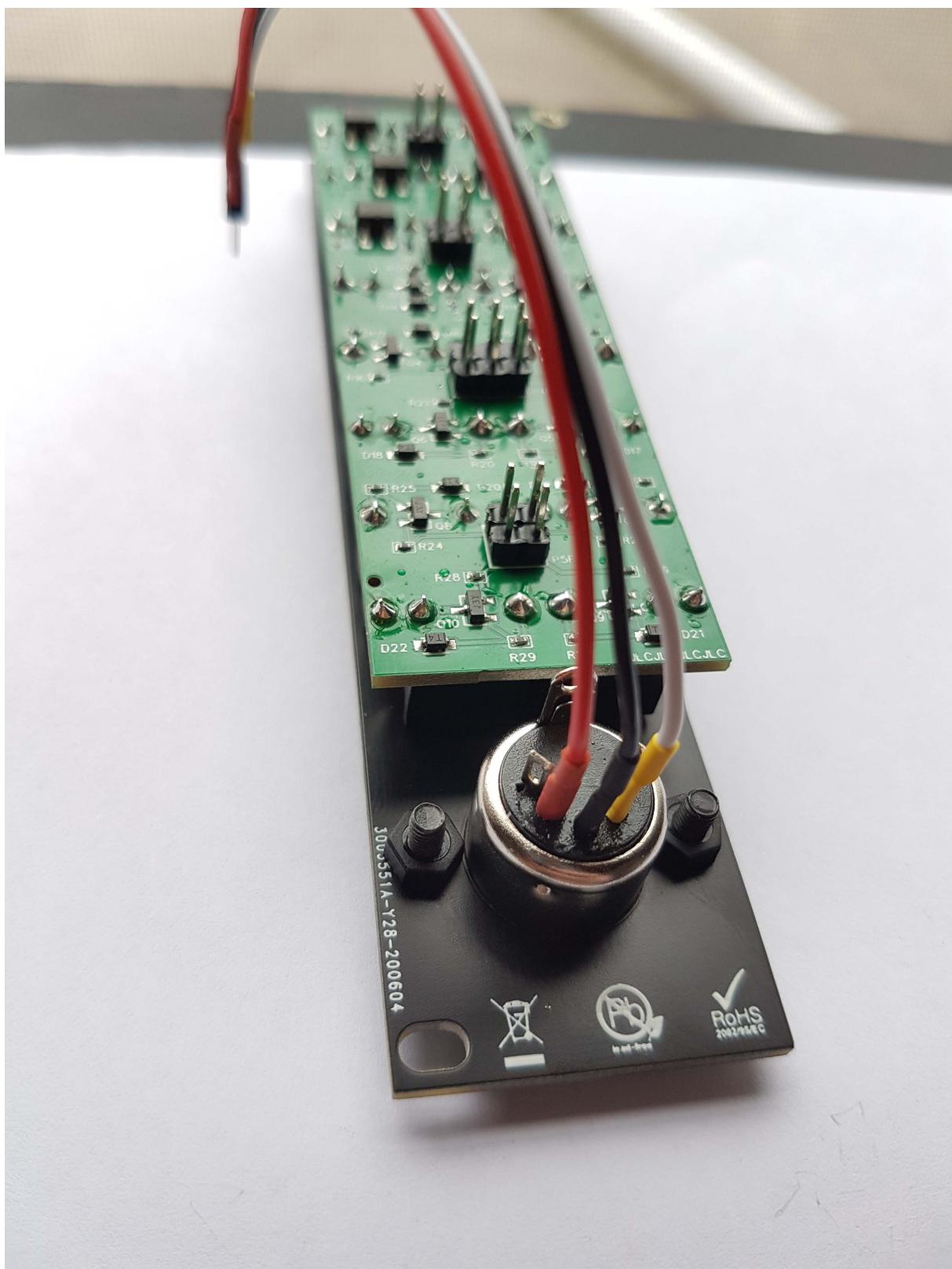


For this step you will need:

- The faceplate with the PCBs mounted on it
- The previously prepared MIDI connector
- 2x hex screw
- 2x nylon nuts

7.1 Install the MIDI connector in the faceplate

Screw it tightly but not too much so you don't ruin the faceplate or the nylon nuts.

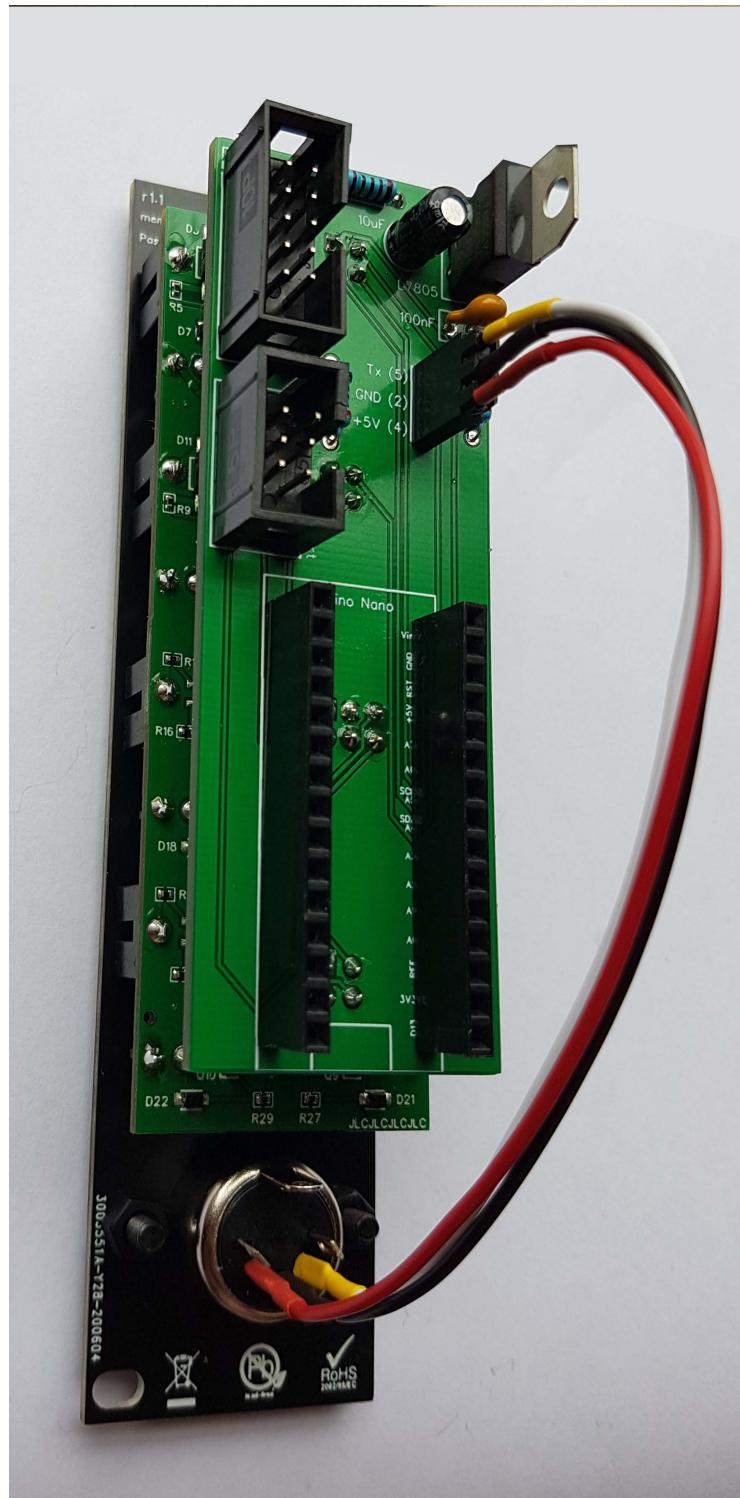


7.2 Connect the MIDI connector to the PCB with the Arduino

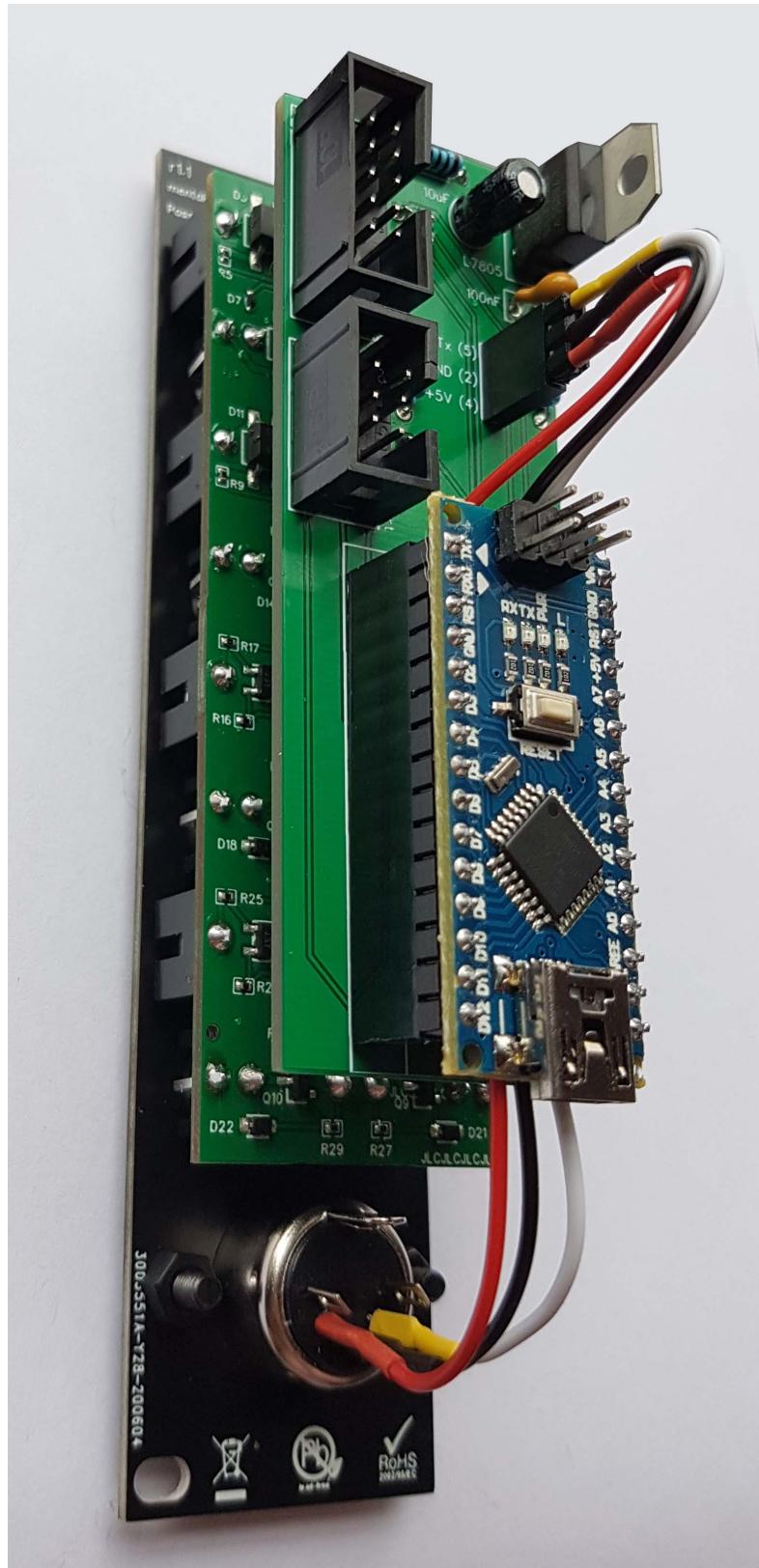
The white wire goes to *Tx* (5)

The black wire goes to *GND* (2)

The red wire goes to *+5V* (4)



7.3 Install the Arduino on top of the wires (or not, it just looks neater to me)



The module is done, time to upload the firmware!

Firmware upload

For now, the firmware is specifically made to control the Korg Volca Beats drum kit but if you want to control another instrument, it is quite easy to modify it to fit your needs.

If you need help adapting the firmware to your needs, feel free to contact me on reddit:
<https://www.reddit.com/user/atulrnt>

1. Download the Arduino IDE

The Arduino IDE is the software we will use to upload the firmware on the Arduino.

Go on <https://www.arduino.cc/en/Main/Software> and download the version compatible with your system.

Once the download is complete, install the Arduino IDE.



2. Download the firmware on Github

Go on <https://github.com/atulrnt/mental-noise-synapse> and click the “Code” button and “Download ZIP”.

The screenshot shows the GitHub repository page for 'atulrnt / mental-noise-synapse'. At the top, there's a navigation bar with links to 'Why GitHub?', 'Team', 'Enterprise', 'Explore', 'Marketplace', 'Pricing', a search bar, and 'Sign in'. Below the header, the repository name 'atulrnt / mental-noise-synapse' is displayed, along with statistics: 2 Watchers, 2 Stars, and 0 Forks. A 'Code' button is highlighted in green, and a 'Download ZIP' button is visible below it. The main content area shows a file tree with files like 'schematics', '.gitignore', 'LICENSE', 'README.md', and 'volca-beats-cv2midi.ino'. On the right side, there's an 'About' section with a brief description of the project: 'Arduino based CV to MIDI module to control the Korg Volca Beats from a modular synth or other source of CV signals'. Below this, there are several tags: 'synthesizer', 'synth', 'modular', 'eurorack', 'eurorack-diy', 'eurorack-synth', 'diy', 'korg', 'korg-volca', and 'volca-beats'. A 'Join GitHub today' call-to-action is also present.

When the download is complete, unzip the downloaded file wherever you want.

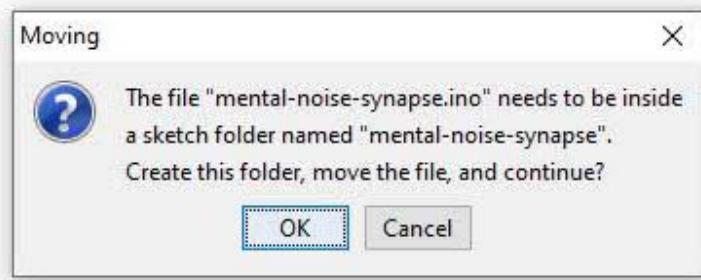
3. Open the firmware

Go into the directory created during the unzip and double click on the file named “mental-noise-synapse.ino”.

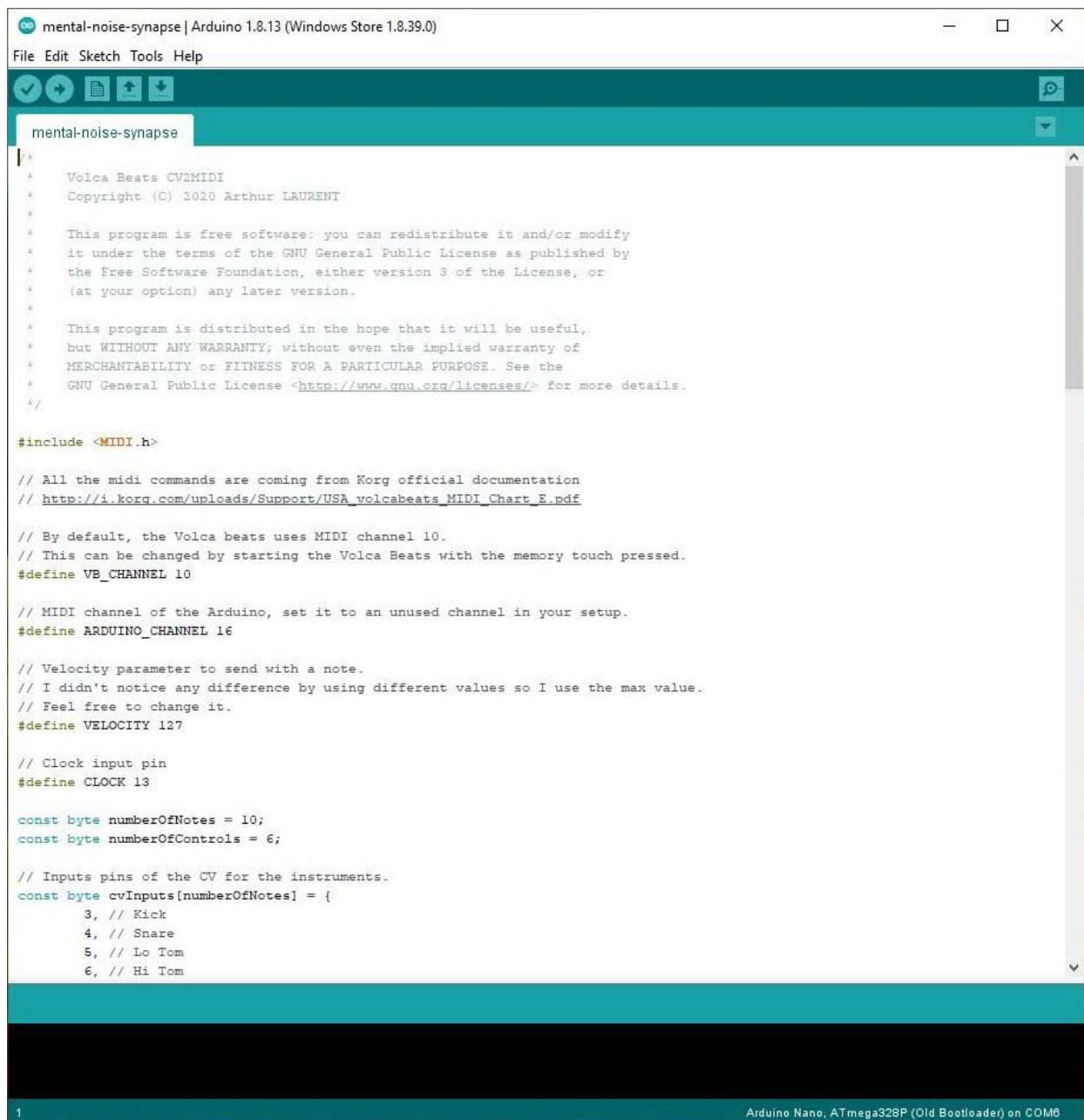
Name	Date modified	Type	Size
schematics	08/06/2020 17:38	File folder	
.gitignore	08/06/2020 17:38	Text Document	1 KB
LICENSE	08/06/2020 17:38	File	35 KB
mental-noise-synapse.ino	08/06/2020 17:38	INO File	5 KB
README.md	08/06/2020 17:38	MD File	4 KB

There might be a message asking you if you want to move this file in a new folder, click “OK”.

Name	Date modified	Type	Size
schematics	08/06/2020 17:38	File folder	
.gitignore	08/06/2020 17:38	Text Document	1 KB
LICENSE	08/06/2020 17:38	File	35 KB
mental-noise-synapse.ino	08/06/2020 17:38	INO File	5 KB
README.md	08/06/2020 17:38	MD File	4 KB



The Arduino IDE should open with the firmware.



The screenshot shows the Arduino IDE interface with the title bar "mental-noise-synapse | Arduino 1.8.13 (Windows Store 1.8.39.0)". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for new file, open file, save file, upload, and download. The main window displays the C++ code for the "mental-noise-synapse" sketch. The code includes comments about the Volca Beats CV2MIDI library, copyright information, and the GNU General Public License. It defines constants for MIDI channels (VB_CHANNEL 10, ARDUINO_CHANNEL 16), velocity (VELOCITY 127), and clock input pin (CLOCK 13). It also defines the number of notes and controls, and provides an array of CV input pins for instruments (3, 4, 5, 6). At the bottom of the code editor, there is a status bar with the text "Arduino Nano, ATmega328P (Old Bootloader) on COM6".

```
mental-noise-synapse | Arduino 1.8.13 (Windows Store 1.8.39.0)
File Edit Sketch Tools Help
mental-noise-synapse
/*
  Volca Beats CV2MIDI
  Copyright (C) 2020 Arthur LAURENT

  This program is free software: you can redistribute it and/or modify
  it under the terms of the GNU General Public License as published by
  the Free Software Foundation, either version 3 of the License, or
  (at your option) any later version.

  This program is distributed in the hope that it will be useful,
  but WITHOUT ANY WARRANTY; without even the implied warranty of
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
  GNU General Public License <http://www.gnu.org/licenses/> for more details.

*/
#include <MIDI.h>

// All the midi commands are coming from Korg official documentation
// http://i.korg.com/uploads/Support/USA\_volcabeats\_MIDI\_Chart\_E.pdf

// By default, the Volca beats uses MIDI channel 10.
// This can be changed by starting the Volca Beats with the memory touch pressed.
#define VB_CHANNEL 10

// MIDI channel of the Arduino, set it to an unused channel in your setup.
#define ARDUINO_CHANNEL 16

// Velocity parameter to send with a note.
// I didn't notice any difference by using different values so I use the max value.
// Feel free to change it.
#define VELOCITY 127

// Clock input pin
#define CLOCK 13

const byte numberOfNotes = 10;
const byte numberOfControls = 6;

// Inputs pins of the CV for the instruments.
const byte cvInputs[numberOfNotes] = {
    3, // Kick
    4, // Snare
    5, // Lo Tom
    6, // Hi Tom
}

1
Arduino Nano, ATmega328P (Old Bootloader) on COM6
```

4. Configure the Arduino IDE to use the right Arduino board

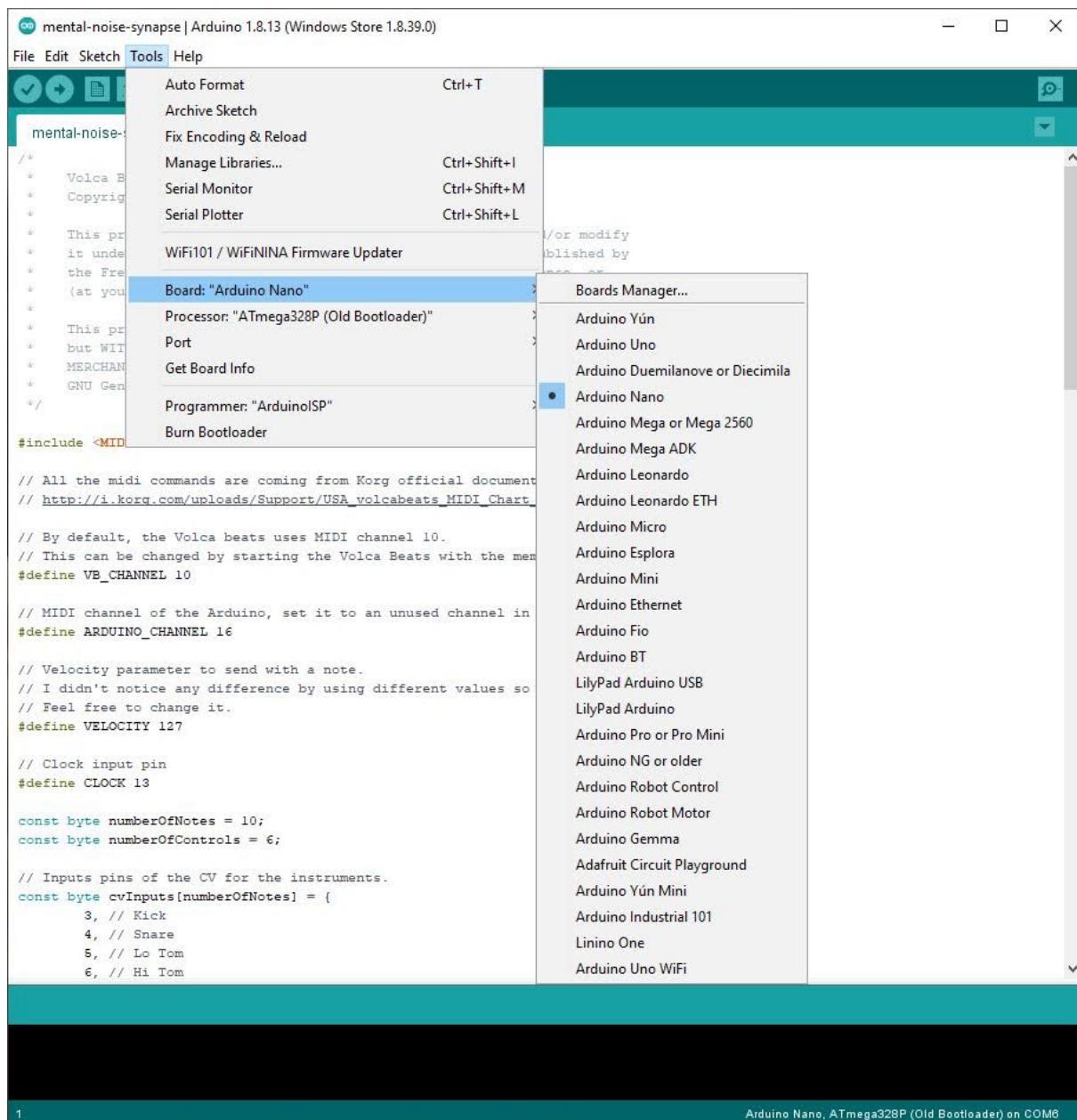
Connect the Arduino Nano to your computer using the provided USB cable.

The Arduino Nano can be plugged to the computer while installed on the module but **do not connect it to the Eurorack while plugged to the computer.**

Always disconnect the Eurorack cable before plugging the Arduino to your computer.

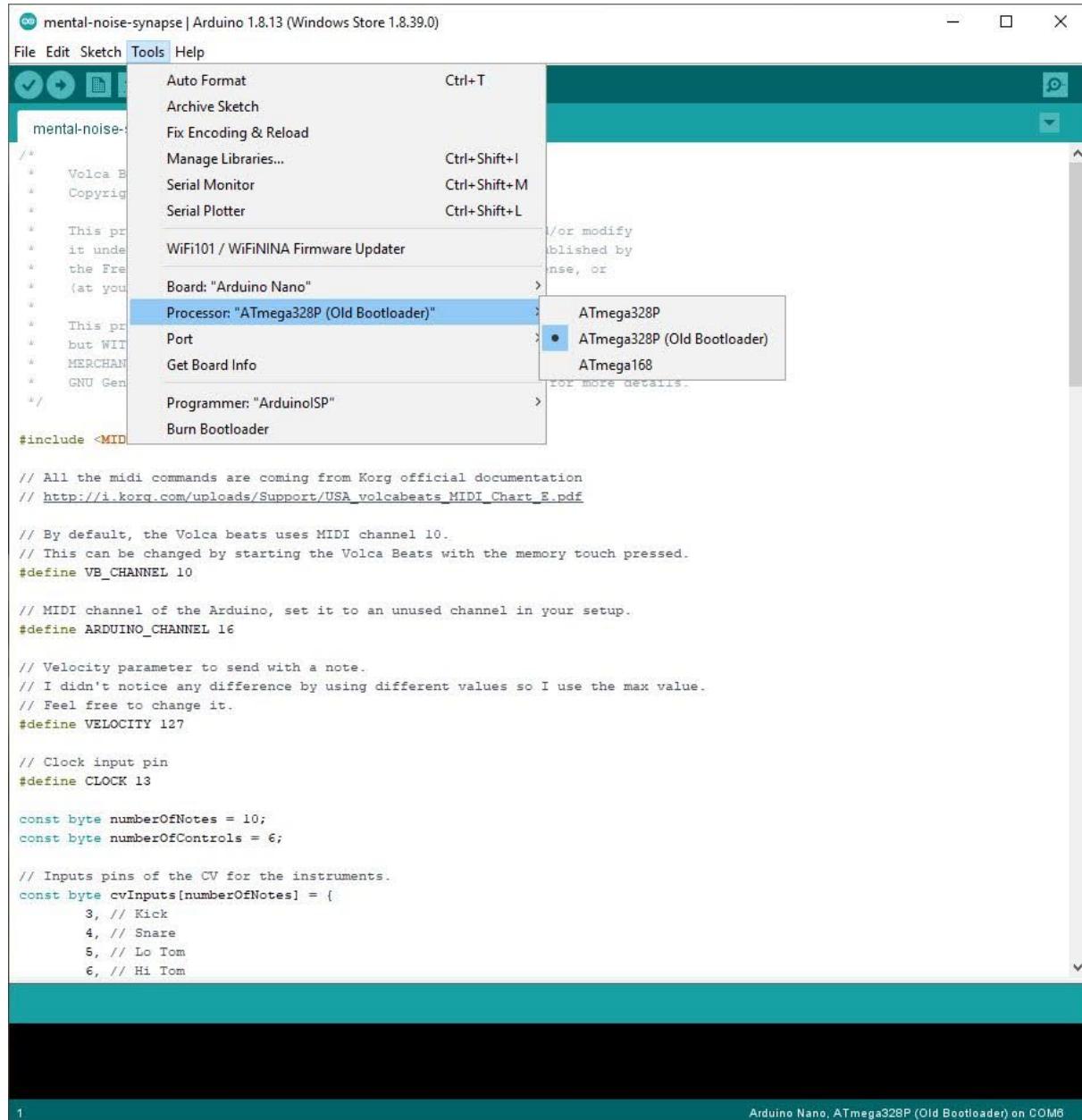
4.1 Choose the right board

In the “Tools” menu, go in the “Board” submenu and choose the “Arduino Nano” in the list.



4.2 Choose the right processor

In the “Tools” menu, go in the “Processor” submenu and choose “ATmega328P (Old Bootloader)”.

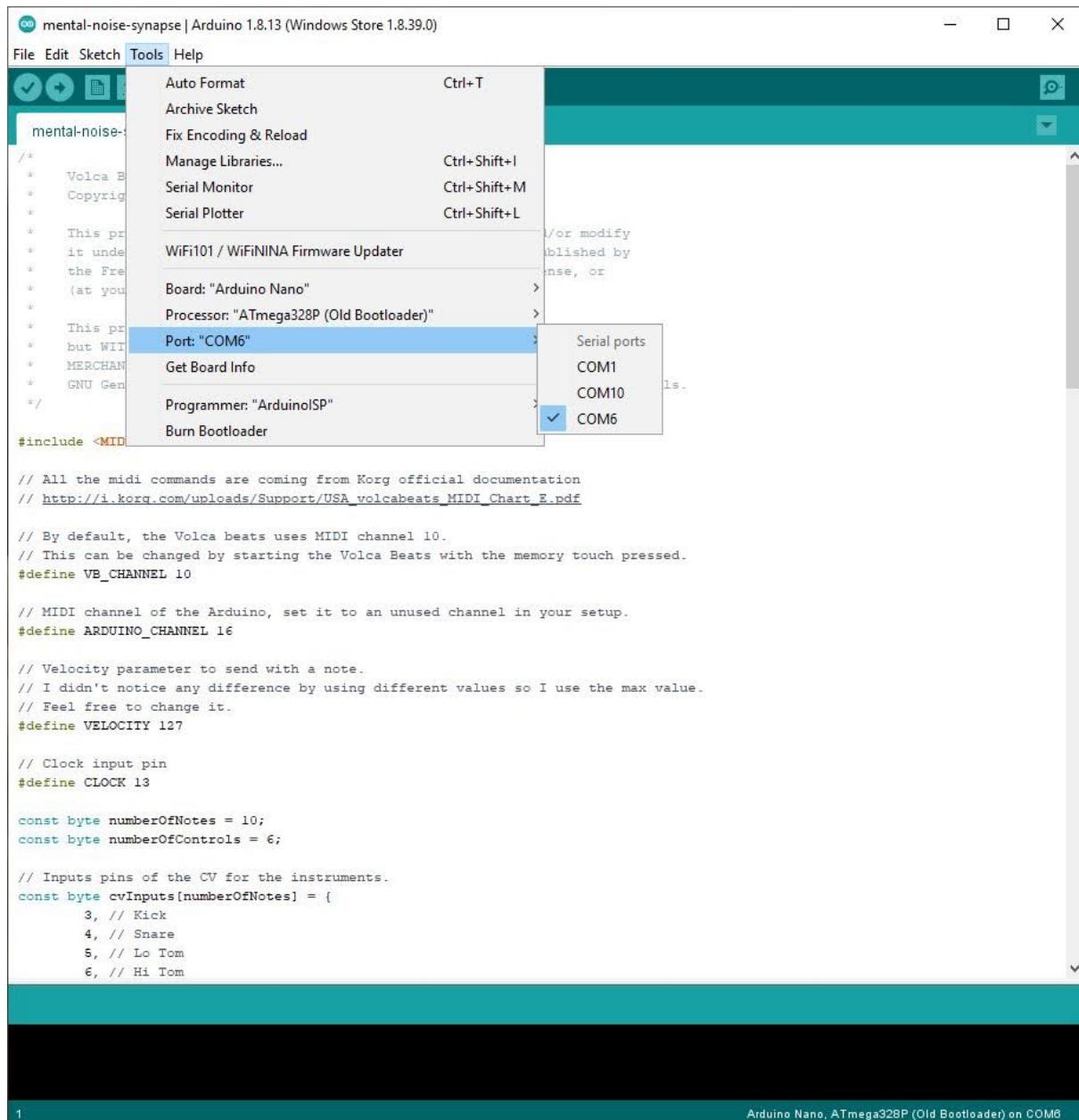


4.3 Choose the right port

In the “Tools” menu, go in the “Port” submenu and choose the port corresponding to your Arduino Nano.

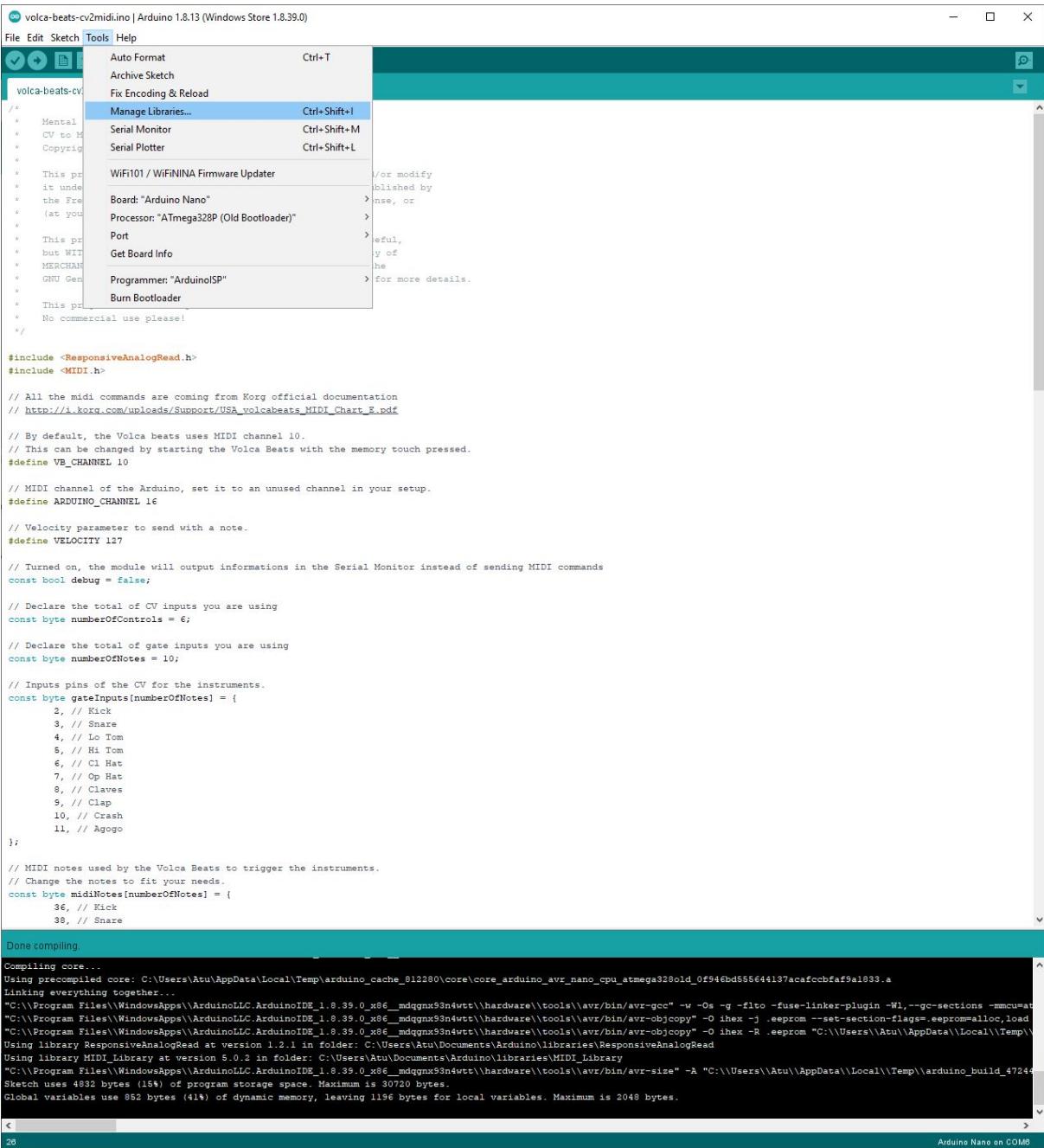
If you don't know, chances are it will be the last one in the list.

If the upload doesn't work in the next steps, just try choosing another port in this list.



5. Add the required libraries in the Arduino IDE

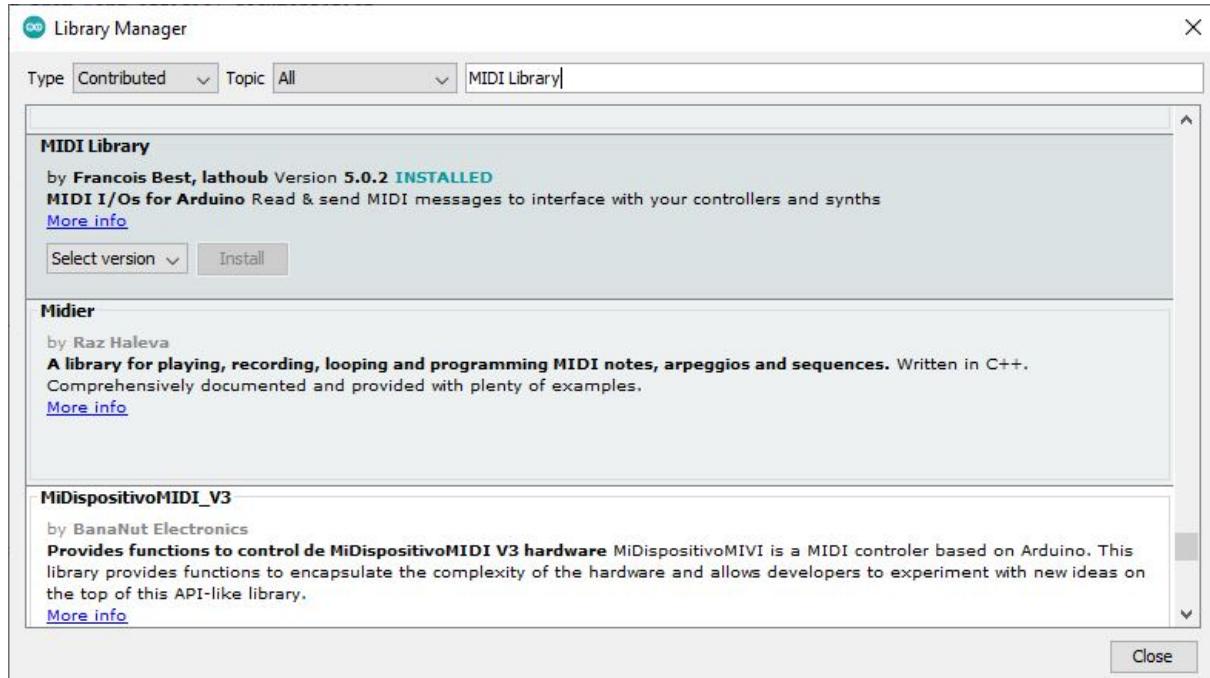
In the “Tools” menu, go in the “Manage libraries...” submenu.



The screenshot shows the Arduino IDE interface. The title bar reads "volca-beats-cv2midi.ino | Arduino 1.8.13 (Windows Store 1.8.39.0)". The "Tools" menu is open, and the "Manage Libraries..." option is highlighted. The main code editor window displays the "volca-beats-cv2midi.ino" sketch. The code includes comments about MIDI commands and instrument mapping. At the bottom of the code editor, there is a terminal window showing the compilation process for the Arduino Nano. The terminal output includes paths like "C:\Program Files\WindowsApps\ArduinoLLC.ArduinoIDE.1.8.39.0_x86_mdggnx93n4wt\hardware\tools\avr\bin\avr-gcc" and command-line arguments for linking and memory allocation. The status bar at the bottom right indicates "Arduino Nano on COM8".

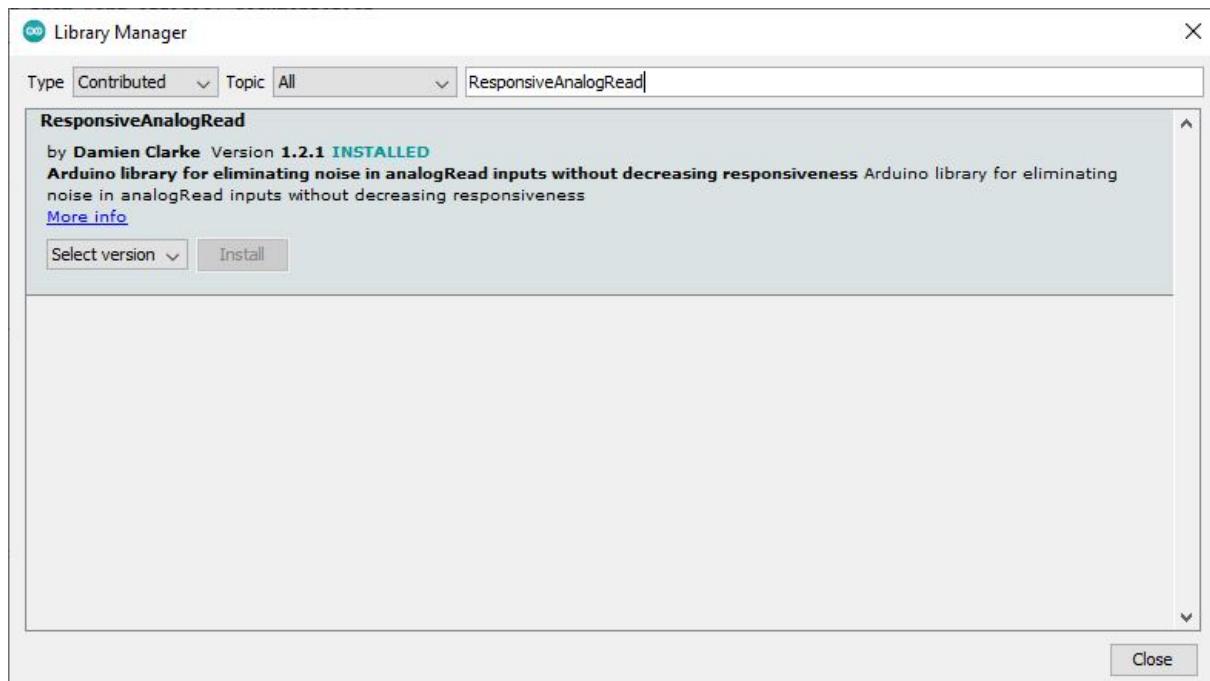
Add the “MIDI Library” by selecting the type “Contributed” and search for “MIDI Library”, you will have to scroll in the list of results to find it.

Click on install to install the library in the IDE.



Add the “ResponsiveAnalogRead” library by selecting the type “Contributed” and search for “ResponsiveAnalogRead”.

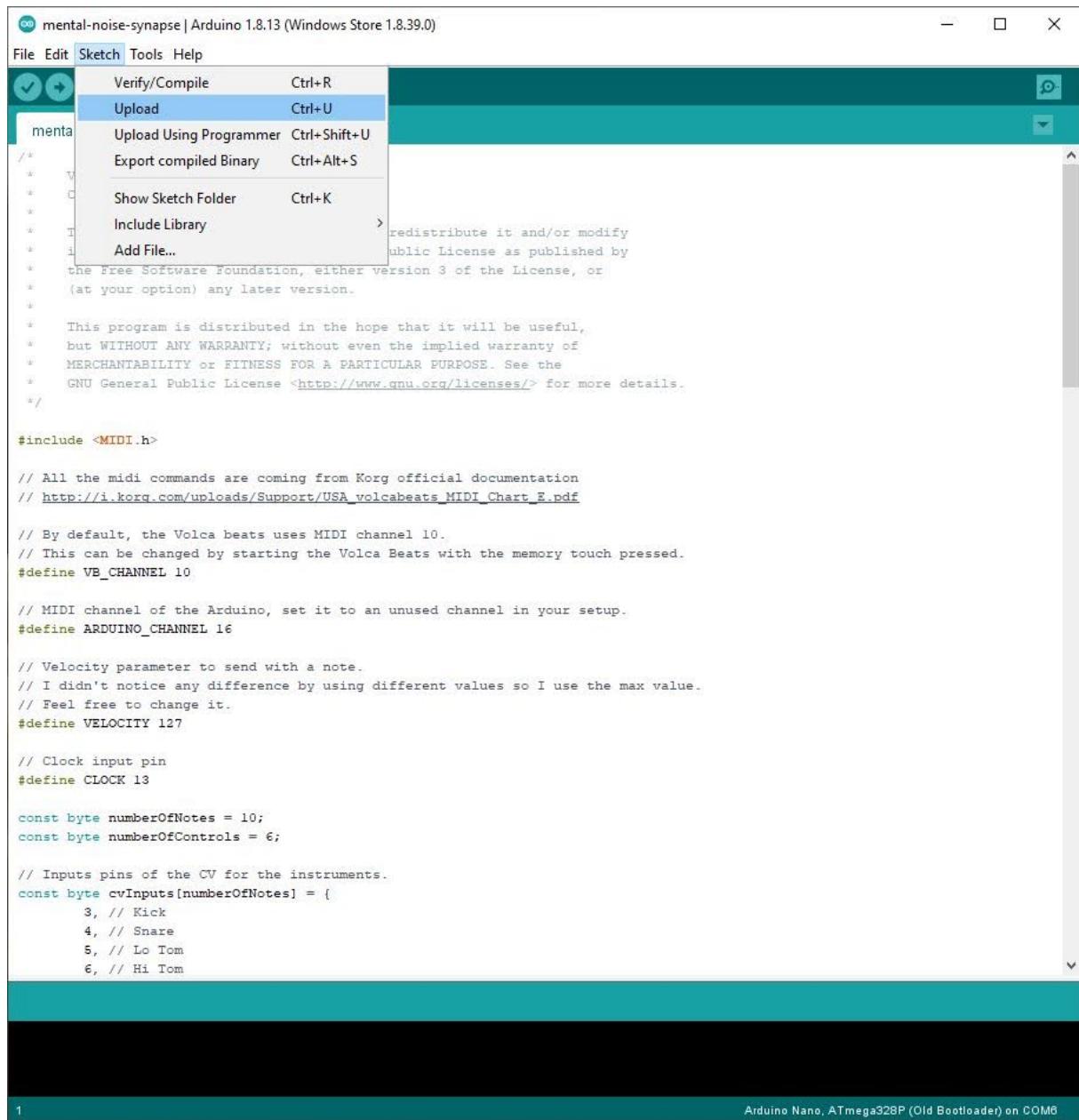
Click on install to install the library in the IDE.



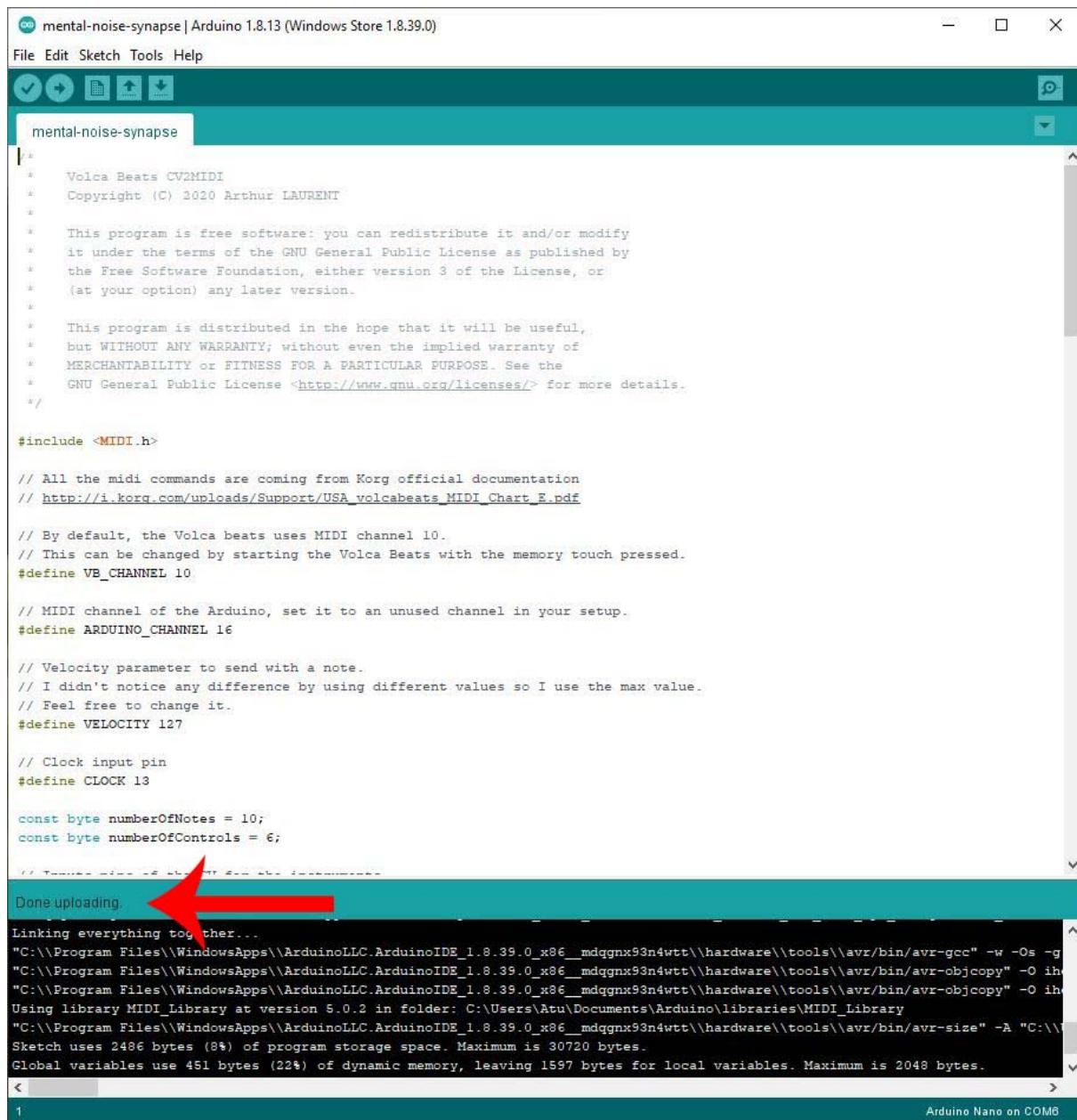
6. Upload the firmware

In the “Sketch” menu, click on “Upload”.

During the firmware upload, some text will appear in the bottom of the Arduino IDE and the Arduino Nano will have its LEDs flashing.



When the upload is done, the Arduino Nano will stop flashing and the text “Done uploading.” will appear in the bottom of the Arduino IDE.



The screenshot shows the Arduino IDE interface with a sketch titled "mental-noise-synapse". The code is a C++ program for an Arduino, specifically for interfacing with a Volca Beats CV2MIDI device. It includes comments explaining the setup and configuration, such as defining MIDI channels and velocity parameters. A red arrow points to the status bar at the bottom of the IDE, which displays the text "Done uploading".

```
/*
 * mental-noise-synapse | Arduino 1.8.13 (Windows Store 1.8.39.0)
 * File Edit Sketch Tools Help
 * 
 * mental-noise-synapse
 * 
 * Volca Beats CV2MIDI
 * Copyright (C) 2020 Arthur LAURENT
 * 
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 * 
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License <http://www.gnu.org/licenses/> for more details.
 */
 
#include <MIDI.h>

// All the midi commands are coming from Korg official documentation
// http://i.korg.com/uploads/Support/USA\_volcabeats\_MIDI\_Chart\_E.pdf

// By default, the Volca beats uses MIDI channel 10.
// This can be changed by starting the Volca Beats with the memory touch pressed.
#define VB_CHANNEL 10

// MIDI channel of the Arduino, set it to an unused channel in your setup.
#define ARDUINO_CHANNEL 16

// Velocity parameter to send with a note.
// I didn't notice any difference by using different values so I use the max value.
// Feel free to change it.
#define VELOCITY 127

// Clock input pin
#define CLOCK 13

const byte numberOfNotes = 10;
const byte numberOfControls = 6;

// Done uploading
Linking everything together...
"C:\Program Files\WindowsApps\ArduinoLLC.ArduinoIDE_1.8.39.0_x86_mdqgnx93n4wtt\hardware\tools\avr\bin\avr-gcc" -w -Os -g
"C:\Program Files\WindowsApps\ArduinoLLC.ArduinoIDE_1.8.39.0_x86_mdqgnx93n4wtt\hardware\tools\avr\bin\avr-objcopy" -O ihex -R .eeprom
"C:\Program Files\WindowsApps\ArduinoLLC.ArduinoIDE_1.8.39.0_x86_mdqgnx93n4wtt\hardware\tools\avr\bin\avr-objcopy" -O ihex
Using library MIDI_Library at version 5.0.2 in folder: C:\Users\Atu\Documents\Arduino\libraries\MIDI_Library
"C:\Program Files\WindowsApps\ArduinoLLC.ArduinoIDE_1.8.39.0_x86_mdqgnx93n4wtt\hardware\tools\avr\bin\avr-size" -A "C:\\" Sketch uses 2486 bytes (8%) of program storage space. Maximum is 30720 bytes.
Global variables use 451 bytes (22%) of dynamic memory, leaving 1597 bytes for local variables. Maximum is 2048 bytes.
< >
1
Arduino Nano on COM8
```

Time to patch!

Congratulations! Your module is now ready to use, you can go ahead and install it in your rack using the provided Eurorack cable.

The Gates inputs are supposed to receive trigger or gate signal from 0 to +5V such as triggers from a clock module to trigger notes or instruments on your MIDI device.

The CV inputs are supposed to receive any CV signal from 0 to +5V such as an LFO output to control some parameters on your MIDI device.

A good idea to start is to put a gate signal in each Gate input, one at the time and make sure it correctly triggers the note or instrument you expect on your MIDI device.

I hope you will find this module useful and you will enjoy using it!

I personally had fun making it, I've put lot of time and efforts into it and I'm very happy with the result.

If you have any remarks or ideas to improve it, feel free to contact me on reddit :
<https://www.reddit.com/user/atulrnt>