



Manual Joystick controller for MPCNC



arminth

[VIEW IN BROWSER](#)

updated 5. 10. 2022 | published 5. 10. 2022

Summary

Update October 2022: Revised the case design slightly. Inforced screwin-bolts, added clearances for LED and buttons, adj

[Hobby & Makers](#) > [Tools](#)

Tags: [arduinonano](#) [joystick](#) [mpcnc](#) [mpcncaddon](#)

Update October 2022:

Revised the case design slightly. Inforced screwin-bolts, added clearances for LED and buttons, adjusted joystick-opening. Version 2b. Version 2 still in for reference

Update November 2019:

I made a PCB for the joystick. So I added the EasyEDA JSON file, the Gerber-files, circuit diagram and new case STL.

If you are interested in a PCB and are located in Europe, please contact me. I have a couple left. Any money I make will go to charity (Paulinchen e.V., an organization helping children who suffered burn injuries).

Update August 2019:

jogdial_v0.3.ino

New features upon request:

Implement treshold for joystick zero for x and y

Swapped order of movements (higher first)

Make 100mm mode configurable individually or disable completely

You have to change these user configurable options in the .ino script (lines 12-15):

```
const int treshold_x = 200; // increase this value, if x-axis of joystick is not giving zero values when idle. Keep values between 300 and 0. Decrease to get joystick act more sensible.
```

```
const int treshold_y = 200; // increase this value, if y-axis of joystick is not giving zero values when idle. Keep values between 300 and 0. Decrease to get joystick act more sensible.
```

```
const long baud_rate=250000; // Baudrate of your MPCNC-Controller
```

```
String _100mm_mode = "100"; // To adjust 100mm-mode individually set to any value eg. "100", "50.5", "20" or set to "disabled" to disable 100mm mode for x and y. Defaults to "100"
```

Update May 2019

I think I have found the issue with the jumps in position. Analyzes showed that these "jumps" were actually correct GCODE commands which would not point to noise by any means. It turned out that while uploading the GCODE remotely thru octoprint to the SDCARD (which took longer than I thought and there was no progressbar showing on the display itself) I was playing around with the joystick and did some homing and movements. Actually these commands not only made it to the machine but were also inserted into the stream of GCODE that was written to the SDCARD. In fact I found these commands in my files on SDCARD! Mystery solved! No issue with wiring or noise - just stupid operator error! So enjoy your joystick (when youre not uploading to your SD!!).

Update March 2019

After a couple of days, I realized some strange jumps of position (x,Y and Z) midmilling. I have a long cable attached to the serial port and I assume these jumps result from noise coupling into the datalines. So please use shortest cables possible with grounded shield. If you still experience strnge things happening (like going to 0,0 while milling and the continue as if nothing happend), disconnect before actually going into production.

And please leave remarks in the comments so I can see if it's just me or others are affected as well.

///

Update as of February 10, 2019

I encountered an incompatibility issue with octoprint (and potentially any other server attached to the USB-port). The controller was blocking the RS232 communication and Octoprint disconnected due to timeout.

I have updated the arduino-script to version v0.2. This should solve the issue - at least it does with my machine! I also updated the file with source-code comments to make it more readable.

version 0 stays in the file for version control purposes only.

If you encounter any problems, please leave a comment.

///

PREWARNING!

You must know what you are doing here! There is a risk of damaging your equipment, if you do something wrong!

Still, it ain't rocket science either!

This is an addon for the MPCNC.

It works only with MPCNC running Marlin firmware on an RAMPS board and compatible like e.g. MKS 1.4, GEN L, Trigorilla or Fysetc F6. Your boards needs to have an AUX-1 port with direct access to the onboard serial port.

BOM:

Arduino Nano 3.0

Joystick module

(e.g. [https://www.aliexpress.com/item/1pc-Free-Shipping-Higher-Quality-Dual-axis-XY-Joystick-Module-PS2-Joystick-Control-Lever-Sensor-For/32656963271.html?](https://www.aliexpress.com/item/1pc-Free-Shipping-Higher-Quality-Dual-axis-XY-Joystick-Module-PS2-Joystick-Control-Lever-Sensor-For/32656963271.html?spm=2114.search0104.3.17.7a7742a5zRLMkv&ws_ab_test=searchweb0_0,searchweb2)

[spm=2114.search0104.3.17.7a7742a5zRLMkv&ws_ab_test=searchweb0_0,searchweb2](https://www.aliexpress.com/item/1pc-Free-Shipping-Higher-Quality-Dual-axis-XY-Joystick-Module-PS2-Joystick-Control-Lever-Sensor-For/32656963271.html?spm=2114.search0104.3.17.7a7742a5zRLMkv&ws_ab_test=searchweb0_0,searchweb2)

R1 680hm

R2 100 Ohm

LED red, green , blue , yello and white

3 tactile switches

piece of breadboard

wiring,

4 pin Dupont female connector

a couple of M3 screws.

PLA for the case

First, fire up the arduino IDE. Load the included script "jogdial_v0.2.ino". Hook up the Nano to your computer's USB. Check the right COM-port in Tools. Switch board to Arduino Nano. Make sure to choose the 328P processor with the right bootloader. If you can't upload, switch to old bootloader (or vice versa). Compile and upload the sketch.

Detach Nano from USB and start wiring. Later, you can still flash a new firmware with the Nano installed in the controller. There is an accesshole in the back to hook up a USB cable.

How to wire you can see on the Fritzing breadboard chart. Make your own board and use my example on the pictures if you want to use my box. Just count the pinholes to have the exact positions of the tactile switches, the nano and the screw- and through holes.

Hook up the 4-pin connector to your lower Aux-1 pin (D0, D1, GND, 5V). Beware! Don't reverse, or it will not work. It may damage your Arduino Nano!!

Make sure, your Marlin Firmware is set to 250000 bps!

If you want to use a different baudrate, change line 57 in the arduino-script accordingly and reupload.

Case design was quick and dirty and by far not perfect. If you want to optimize, please feel free. OpenSCAD file is attached.

Operation:

On power on, system does a lamp-test. All five LED must light up for a second. Then, the blue LED is lighting.

Press HomeXY to home the system. Press Home Z, if you have a tool-height thing attached on your Z-endstop.

If you have set your spindleheight to 0, press reset coord button to zero XYZ.

For manual driving:

Blue LED on:

Using Joystick advances Toolhead in $\pm 0.1\text{mm}$ increments. Keep pushing and movement repeats. You can go diagonal as well!

Push on joystick to switch to 1mm mode (green)

Green LED on:

Using Joystick advances Toolhead in $\pm 1\text{mm}$ increments. Keep pushing and movement repeats. You can go diagonal as well!

Push on joystick to switch to 10mm mode (yellow)

Yellow LED on:

Using Joystick advances Toolhead in $\pm 10\text{mm}$ increments. Keep pushing and movement repeats. You can go diagonal as well!

Push on joystick to switch to 100mm mode (red).

Red LED on: (Red means DANGER!!!)

Using Joystick advances Toolhead in $\pm 100\text{mm}$ increments. Keep pushing and movement repeats. You can go diagonal as well! Be aware not to exceed your machine limits!! When moving 100mm increments, do it one move after the other!! I am not responsible for any damage you do to your machine!! At least I can say, I told you so! ;)

Push on joystick to switch to Z-Axis and to 0.10mm mode (white and blue)

White LED always indicates Z-axis is moving!!

White&Blue LED on:

Using Joystick in Y-direction advances Toolhead up and down in 0.1mm increments. Keep pushing and movement repeats.

Push on joystick to switch to 1mm mode (white green).

Push on joystick to switch to 10mm mode (white yellow).

Push on joystick to switch to 20mm mode (white red).

Again, beware!! Don't exceed your machine's dimensions! Nothing prevents you from driving your spindle into the worktable!!

Push on joystick again to return to XY-Mode 0.1mm again. Repeat!

Pretty simple and fun!

How it works:

The Arduino Nano reads the joystick analog signals and the button signals and turns them into LED status and machine-commands.

The commands (eg. G28 X Y) are physically sent to the onboard serial port (D0, D1). Marlin executes these commands directly.

Enjoy driving your MPCNC with a Joystick and not with the stupid display-menu!

Cheers

Armin

Print Settings

Printer Brand:

Creality

Printer:

Ender 3

Rafts:

No

Supports:

Yes

Resolution:

0.2

Infill:

15%

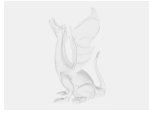
Filament: Noyes PLA

grey Joystick controller MPCNC Operation, Part 1&2

=====

Category: Machine Tools

Model files



large_display_speedopult2_152790.stl



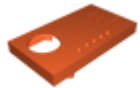
large_display_speedopult2_deckel_152790.stl



speedopult2_deckel.stl



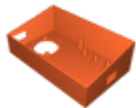
case_bottom.stl



case_top.stl



speedopult2.scad



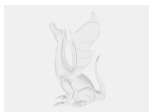
speedopult2.stl



case_top_bottom.stl



case.scad



speedopult2b.scad

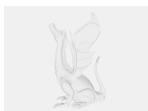


speedopult2b.stl

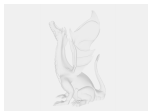
Other files



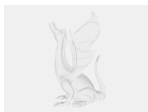
schematic_mpcnc_joystick_circuit_diagram_20191112.pdf



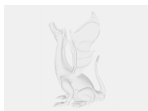
jogdial.ino



how-to-order-pcb.txt



jogdial_v02.ino



jogdial_v03.ino

[Find source .stl files on Thingiverse.com](#)

License

This work is licensed under a
Creative Commons (4.0 International License)



Attribution

- ✗ | Sharing without ATTRIBUTION
- ✓ | Remix Culture allowed
- ✓ | Commercial Use
- ✓ | Free Cultural Works

✓ | Meets Open Definition