Setting Up Limit Homing Switch

Edit New page

bdring edited this page on May 19, 2021 · 16 revisions

Setting up Limit/Homing Switches

Overview

Note: This page is a work in progress and will be added to as time permits and questions arise. Thanks for your patience

This page is designed to help newbies have basic success with homing. If you are a newbie, please start with a basic setup. Don't go with double ended switches, hard limits, etc. in your first go. It will be much easier to debug.

A typical homing sequence works like this. The machine will move towards a switch at a relatively rapid rate. Once it detects the switch it will back off the switch and rehome at a slower rate to get a more accurate home position. It then backs off the switch. The machine work position is then set. Homing can be done one axis at a time or several axes can be done at one time. The default is one axis at a time.

All CNC machine should have homing switches. It makes sure your machine knows where it is in the machine space. If you have an issue or just want to exactly return to a position home switches are required.

Here is a YouTube video I did a while ago that might help you understand some of the concepts.

Switch Types

The default circuit is an I/O pin with an internal pullup resistor. Some pins don't have internal pull ups and will need an external pull up.

Mechanical Switches

A variety of switch types can be used, but generally they will be either normally open (N.O.) or normally closed (N.C.) or have both types of contacts. I tend to recommend normally open switches for newbies. These switches would complete a connection when pushed. This is the default type for Grbl. They are a little more forgiving if you accidentally send an output signal to them when programming. An output signal sent to a closed switch would likely permanently damage the ESP32.

Normally closed are considered more fail safe because a break in wiring will trigger the switch. They are also a little more robust against noise when in the closed state if they close to ground.

Use a switch with good mechanical mounting. It helps to have a little overtravel after actuation because Grbl may over travel a bit in the search phase. It is also a good idea to have a mechanical stop that the axis can crash into if the switch fails to stop the motion.

There are a lot of 3 (3) switches available (amazon, eBay, AliExpress) for 3D printers. One of the wires is a positive voltage, so an LED on the switch will indicate when it is pushed. You can generally use them in a (2) wire mode. Just use a meter to see which wires connect when the switch is pushed. For example the switch shown below would wire the red and white wires the the controller.

Electronic Switches

- FAQ
- Help with Switch Problems
- Requesting Help

Development

- Pull Request Guidelines
- MSG Style Guide

FluidNC (Next Gen)

- About

Testing public edit

Clone this wiki locally

https://github.com/bdring/Grbl_Esp32

These are typically optical or proximity switches. You generally need to provide these with power ground. Be sure they are compatible with the 3.3V max input to the ESP32. Check the output levels in the open and activated state.

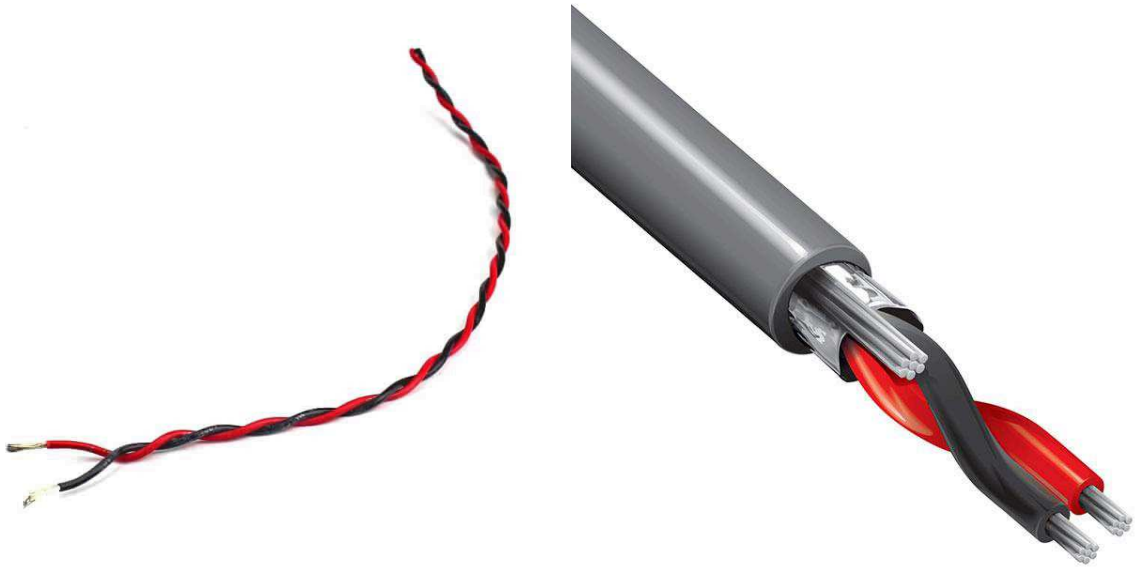
Wiring

...coming soon

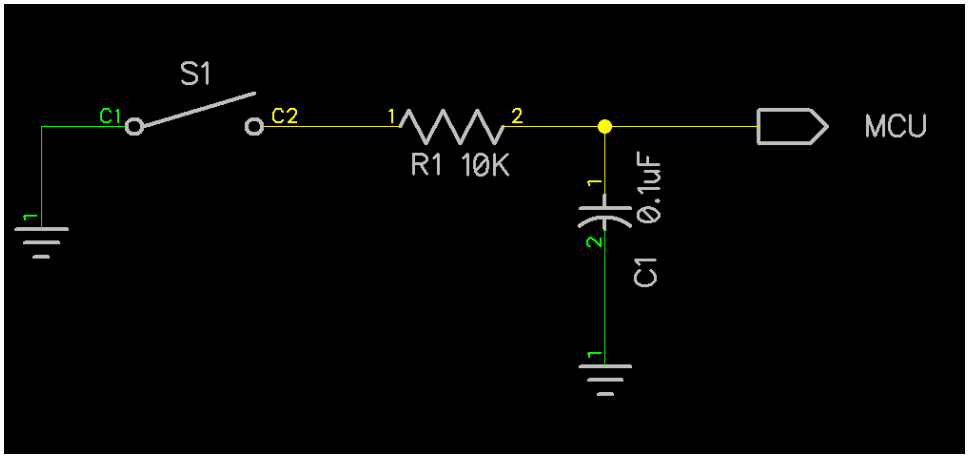
Noise Filtering

Electrical noise can cause false readings. The noise is typically short, low energy spikes that can be reduced in various ways.

Start with good wiring practices. Use a separate signal and ground wire for each switch. Twist that pair of wires at about 1-2 per twist. You could also use shielded wire with the shield ground at the controller side only. Keep wires as short as possible and away from motor wires.



You could add an RC filter to the the switch.



You could enable software debouncing. This causes the firmware to ignore very short pulses. This is enabled with `#define ENABLE_SOFTWARE_DEBOUNCE` in config.h

Optional: Change to normally closed (N.C.) switches. Normally closed switches normally have low impedance connection to ground, which is harder to impart noise to. You could still have a noise issue when they open. You need to change the $5 setting for this. You can use N.C. switches when you use more than one switch per axis by putting them in a single wire loop through those switch. This loop may be more susceptible to noise though.

Machine Definition

You need to map switches to ESP32 pin numbers. You can map up to 2 switches per axis on up to 6 axes. Here are some examples.

```
#define X_LIMIT_PIN             GPIO_NUM_17
#define Y_LIMIT_PIN             GPIO_NUM_4
#define Y2_LIMIT_PIN            GPIO_NUM_2
#define Z_LIMIT_PIN             GPIO_NUM_32
```

If you have 2 inputs per axis, the firmware treats them exactly the same. The status will report "X" if either one or both X_LIMIT_PIN and X2_LIMIT_PIN are active. The second input is used if your switch type requires it or if it makes your wiring easier.

Settings

There is more on the use of these settings in the text below.

- **$Stepper/DirInvert** (This sets the direction of travel of the axes during homing)
- **$Limits/Invert** (This inverts the switch logic for normally open and normally closed switches)
- **$Limits/Soft** (soft limits)
- **$Limits/Hard** (hard limits)
- **$Homing/Enable** ($Homing/Enable=Off disables homing $Homing/Enable=On enables homing)
- **$Homing/DirInvert** (Homing search direction invert bit mask)
- **$Homing/Feed** (The feed rate during the slower locate phase)
- **$Homing/Seek** (The faster search feed rate)
- **$Homing/Debounce** (Switch debounce delay) This sets a little delay between homing phases to let the switch settle.
- **$Homing/Pulloff** (The distance the the machine moves away from a switch after touching it)

Firmware Options

Newbie, please initially stay away from these option

...partial list

- **#define HOMING_INIT_LOCK** If defined Grbl will start in alarm mode until home or the alarm is cleared with **$X**
- **$Homing/Cycle0** (0 through 5) This Setting tells Grbl the order of axes to home and how many to home in each cycle. Examples: $Homing/Cycle0=Z to home Z on the first cycle and $Homing/Cycle1=XY to home X and Y on the second cycle.
- **#define N_HOMING_LOCATE_CYCLE 1** This is the count of pull off re-homes it does for improved accuracy. Typically a value of 1 is used. You can do up to 128. You can also change it to 0 if you just want the initial search to set the home position
- **#define HOMING_SINGLE_AXIS_COMMANDS** This enables the $HX, $HY single access homing commands. Grbl_ESP32 has this enabled by default. If you have a machine like a delta, you may want to disable this.
- **#define HOMING_FORCE_SET_ORIGIN** Normally Grbl sets it's machine work area in negative space (not defined), so if you have a 300mm X axis max travel and that axis homes in the negative direction. After homing your position will be -300 (plus whatever you pull off value is). This throws off a lot of newbies. They want their machine to be zero'd after homing. In reality you don't zero in machine space you zero work spaces like G54. (watch the video)
- **#define LIMITS_TWO_SWITCHES_ON_AXES** This allows you to put a limit switch on both ends of the axis. They are connected to the same I/O pin. This lets Grbl know that if a switch is activated before homing, there is no way to know which way to back off the switch.
- **#define INVERT_LIMIT_PIN_MASK**
- **#define CHECK_LIMITS_AT_INIT**
-

First Tests

I strongly suggest doing your initial testing from the USB/Serial port. It is much easier to see all the original messages and status.

Switch Activation

The first thing to do it to check that the switches are reporting correctly. Manually move the machine axes so no switches are being activated by an axis. Send **?** to get the current status. It should report something like this.

```
<Alarm|WPos:0.000,-0.000,0.000|Bf:15,128|FS:0,0>
```

Now manually activate one of the switches. It should look something like this.

```
<Alarm|WPos:0.000,-0.000,0.000|Bf:15,128|FS:0,0|Pn:X>
```

The thing you are looking for is the **Pn:X** (or whatever axis you activated) part of the status. You can ignore everything else. You need to see no Pn: in your status without switches push and the correct Pn: status for each axis switch activated. Pn: will report the XYZABC axis letters and also PDRHS (Probe,Door,Reset,Hold,Start). You want to make sure **no switches** are reporting when the limit are not activated.

If you see all switches reporting in the Pn: status when no switches are being activated something is inverted. The $5 setting inverts the switch reporting status. $5 can be either $5=0 or $5=1. Swap the $5 value if you are not seeing the correct status. Recheck the status of no switch activated and all switches individually activate. Do not move on the the next step until you are getting correct status.

Axis Movement Directions

The next thing to do is check that the axes move in the right direction. If you send it a move in the positive direction, does the axis move in the positive direction. Here is a sequence I like to use.

First send **$X** to clear any alarms. Send **G91** to put the machine in incremental mode. The machine will just respond with **OK**, and not move. Now each time you send something like **G0X2** it will move 2 units in the positive direction. If you send **G0X-2** it should move 2 units in the negative direction. Use a short move for each axis to make sure they all move in the correct direction.

If any axes move in the wrong direction, there are 2 ways to fix this. The first is with the **$Stepper/DirInvert** setting. The setting is a list of the axis direction you want to invert. For Example: **$Stepper/DirInvert=XZ** means X and Z are currently inverted. Check the current setting by sending **$Stepper/DirInvert** and then add or remove letters to change the direction of the axes.

The other way to to change the stepper motor wiring. You would swap the wires on **only one** coil of the motor going the wrong way. While this is a little more work, **Note:** on machines with 2 motors on an axis, fixing the wiring is the only way to fix the direction.

Homing Direction

The axes can home in any direction you prefer. For example: it is common on a lot of routers to home X and Y in the negative direction and Z in the positive (up) direction. This is controlled with the **$Homing/DirInvert** setting.

Homing Cycles

The axes are homed in cycles. Typically a Z would home up first to clear the work before you home in X and Y. You can have more than one axis home per cycle. You might want to home Z on the first cycle and then home X and Y at the same time on the second cycle. When setting up a machine it is probably best to do one axis per cycle. Once everything is working, you might want to speed things up with multi-axis homing. You can have up to 6 cycles. Examples set **$Homing/Cycle0=Z** to home the Z on the first cycle. Set **$Homing/Cycle1=XY** to home X and Y together on the second cycle. Set **$Homing/Cycle2=** to put nothing on the third cycle. Make sure all cycles have what you want.

Homing Test Setup

Make sure the following settings are in right.

- **$Limits/Soft=Off** (Turn of soft limits)
- **$Limits/Hard=Off** (Turn off hard limits)
- **$Homing/Enable** (Enable homing)
- **$Homing/Feed=100** (a slow homing feed rate)
- **$Homing/Seek=200** (a faster search rate)
- **$Homing/Pulloff=3** (set homing switch pull off to 3mm)
- **$Homing/Cycle0=Z $Homing/Cycle0=X $Homing/Cycle0=Y** (or whatever you need, but one axis per cycle)

Homing Test

Homing is done using the **$H** command. By default you can also use single axis homing with **$HX**, etc. Single axis homing is good for testing, so I will assume it is enabled

Be ready to kill the power or hit the ESP32 reset button. Send **$HZ**. The axis should move towards the limit switch, touch it, back off and repeat at a slower speed.

If it just does a short slow move and issues an **ALARM 8**. It means Grbl_ESP32 detected an activated switch before homing and tried unsuccessfully to clear the switch by backing off. Note: If you have switches at both ends of the axis (set in firmware) Grbl will not know which way to back off and immediately issue the **ALARM 8**.

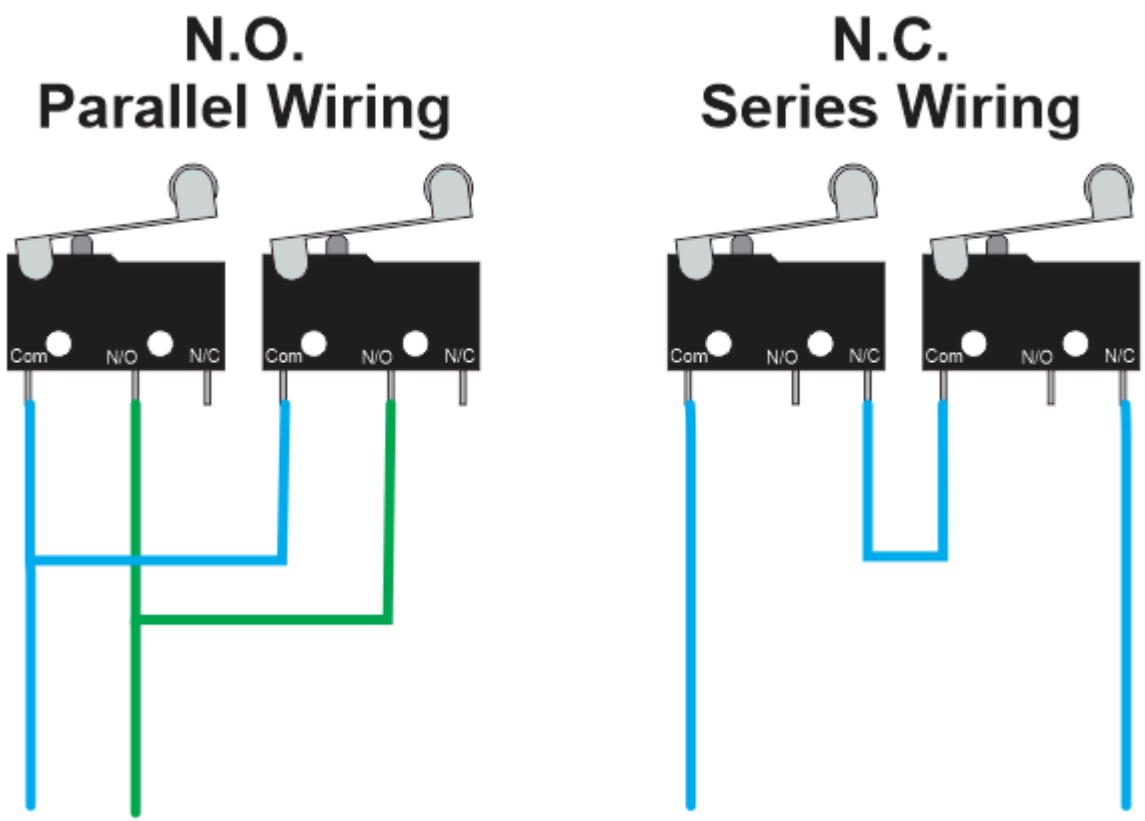
Tuning

Once you have basic homing working, you can tune some values to get better performance. Make sure the $27 pull off is fully clearing the switch. Play with the $24 and $25 speeds. It is nice to have a relatively quick search phase followed by a slow second locate phase,

Advanced Topics

Dual Switches On One Input

You can put switches on both ends of your axis. They would connect to the same input pin. If you are using N.O. switches, they should be wired in parallel. If you use N.C. switches they should be wired in series. Most forms of electronic switches probably cannot be used in this mode.



Soft Limits

Soft limits, **$Limits/Soft=On**, use Grbl's internal position values to determine if a requested move will crash into one of the ends. It uses the max travel ($13x) to determine this. The machine will enter an **Alarm 2** mode. You must reset the machine, but no position has been lost and you do not need to rehome. If you want soft limits to ignore an axis, set that MaxTravel on that axis to 0.

Hard Limits

Hard limits, **$Limits/Hard=On**, use the switches to stop Grbl if it hits a limit switch. This is a good fail safe setup, but it does an immediate uncontrolled stop. You must rehome before you can use the machine again. Also, CNC machines tend to create a lot of electrical noise when running and can cause false limit switch triggers. Test your machine a lot before using this mode.