

Shop Dust Collection With Motorized Gates and Touchscreen Control



by imageguy

It was finally time to solve my shop dust collection issues once for all. I had a 4 inch 1500cfm dust collector, but was dragging flexible dust hose from machine to machine. A while ago, I saw on Adam Savage's "Tested" a system with motorized gates. Since some of my gates would be under the workbench, I thought it was a lovely idea. You can buy these things, but making your own is more fun.

This Instructable shows how to build the system I came up with. It's based on 4" ID sewer pipes, gates are powered by servo motors and the controller is a cheap ILI9488 touchscreen controlled by an Arduino Nano. Everything runs on 5V.

All of the STL files, model source files, Fritzing project and Arduino sources can be found on my [GitHub](#).

A separate [Instructable](#) describes using ILI9488 touchscreen display. Useful as background, but not required for this project. A separate [Fritzing Parts](#) repository on GitHub contains non-standard parts used in the Fritzing circuit layout.

This is a reasonably complex project, but following this Instructable you should be able to avoid going down various rabbit holes that sucked up so much of my time.

I hope you find this project useful, or at least mildly entertaining.

Supplies:

Dust collection blower, such as this [WEN 3403](#).

5V servo motors, one per gate. I used [MG996R](#) motors.

5mm [stainless steel rods](#), 65 mm or so segment per gate.

70mm by 50mm prototype circuit board.

[Arduino Nano V3.0](#).

3x 3.3V to 5V [logic level converter](#) Be careful, since some converters don't have performance to support SPI, protocol used to communicate with the display.

3.5" ILI9488 touchscreens can be found mounted on several different boards with different pinouts. I used [AliExpress](#) - \$11.5 + \$4.68 shipping, I bought two for \$27.41. The same model can be found on [Amazon](#) for \$20.26.

SPST switch.

4 inch ID sewer pipe and whatever couplers, corners and junctions are necessary.

A smooth board for making gate housings. I used Rubbermaid 11.8"x35.8"x5/8" shelf. Assuming 4" piping, each gate requires two 6x6 inch pieces.

Thin stock, such as laminate or HDPE for the moving part of each gate. Something a little thicker as separators to join two gate housing panels together and leave space for the moving gate.

#6-1/2" wood screws, 3 per gate.

M3 screws (20mm), washers and nuts, to mount motors, 4 per gate.

M3 screw (16mm) to mount a gear to the motor.

M3 screws (35mm) to attach the sliding gate to the mechanism, 2 per gate.

good strong glue (wood glue won't work to attach pipe to gates).

wire to connect the controller to gates.

USB adapter, at least 2A.

Tools

3d printer. I printed all the parts using PLA filament.

Dremel or similar tool.

router

jigsaw

basic woodworking tools.

soldering iron and basic electronics tools.

5mm reamer drill bit comes really handy.



<https://youtu.be/UGEeao8J4jg>

Step 1: Design Considerations

When I started this project, I thought that the gates must be built to fine tolerances so as not to lose the air pressure. This is not the case. The actual gate can fit into the body quite loosely. If the gate is closed, the air pressure will pull the gate to close the outlet. If the gate is open, very little or no air escapes around the gate.

Assuming that the blower is turned off, a gate needs very little power to move. MG996R servos I used are probably an overkill. Most of the resistance to movement comes from plastic gears. I found it helpful to clean them up with a utility knife and 400 grit sandpaper after printing to make the movement easier.

On the other hand, you do need a little resistance in the mechanism. The gates are kept unpowered unless the gate is moving. Resistance keeps the gate in open or closed position even if the gate movement is not horizontal.

Gates can be kept powered at all times by changing the sketch, but you might need a stronger power supply. If only a single gate is powered at the time, a 2A 5V power supply will be enough no matter how many gates are in the system.

The circuit is designed to power 7 gates, since Arduino Nano has 7 unused pins after the display is connected and two pins are kept for a serial connection. The two remaining general purpose pins are marked "analog only" and I don't know if they can be used to drive servos.

The sketch controls six gates, which is what I currently use in my workshop.

I originally got interested in touchscreen for a different, yet-to-be-built device, but it proved to be an excellent fit for this project. I used the development board while developing the gates. It's also nicely flexible, easy to change the labels if I repurpose any of the gates or add one more gate.

Step 2: Gate Hardware

We start by making a jig for making the pipe hole. In a reasonably thick piece of plywood, cut a circle of 4" or so diameter. Mount this over one of pipe fittings and use the fitting surface to smooth the hole with a router. This hole should provide a reasonable fit for a 4" ID pipe.

For each gate, cut two pieces of pipe, 3" or so long. Cut two 6x6" pieces from whatever stock you are using for the gate body. Laminated particleboard shelf I used worked well. You want something reasonably thick, so the pipe sits well and there is enough surface for glue. Mark the circle to cut. You need at least 20mm space towards the opening where the gate will go in and out and at least 40mm to the right (looking from the opening) for the gate mechanism.

I had no idea how much space I'd need, so most of my gates were originally cut much too big - I trimmed the excess later in the process. You don't want the mechanism to be too close to the pipe, since some space is needed to put on the cover and you'll be inserting this pipe into some sort of fitting.

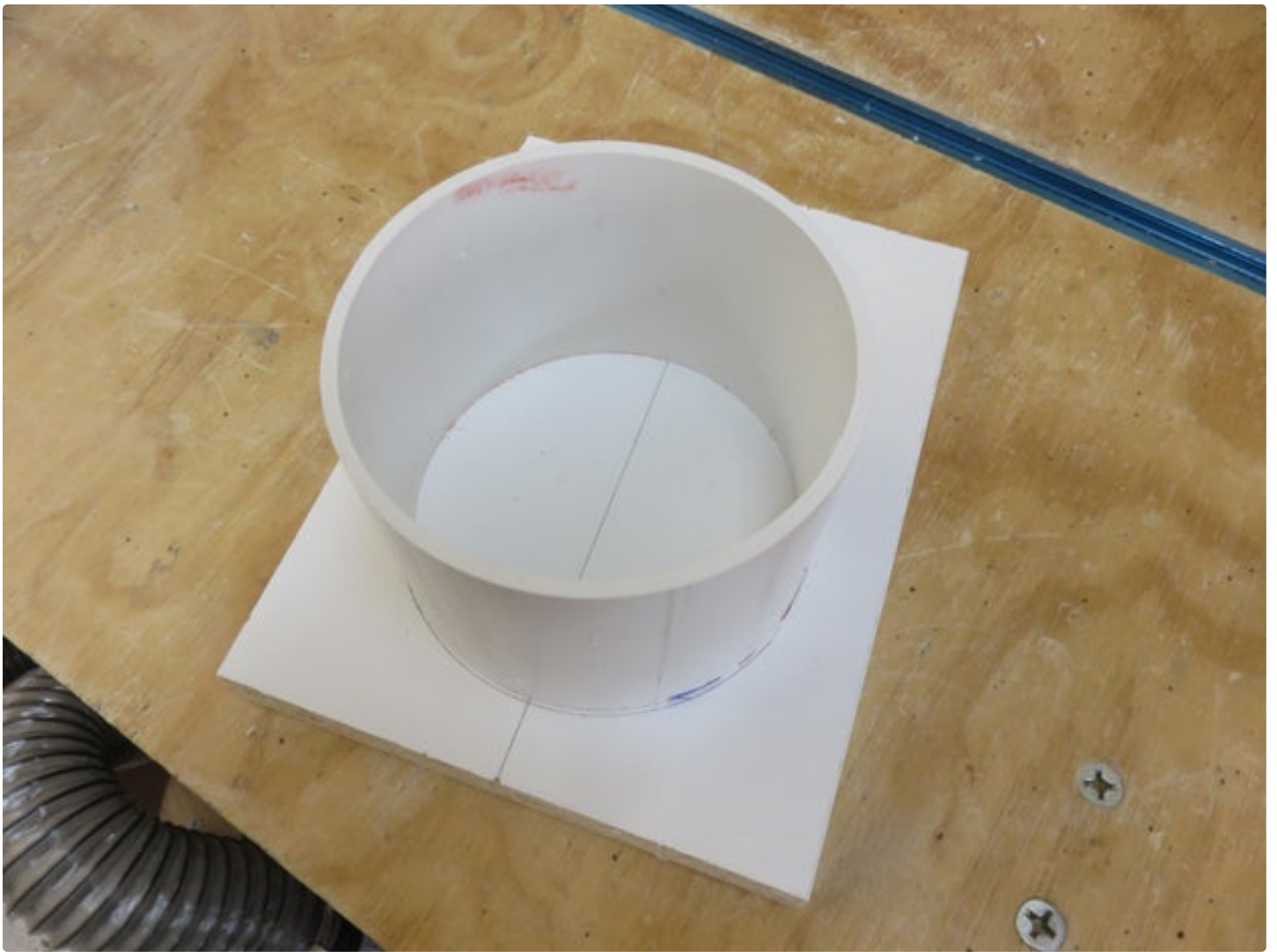
Cut a circle slightly smaller than the mark with a jigsaw (you'll have to drill a hole to start). Attach the jig so the jig hole is centered with the irregular hole cut in the board. I used a nailer. Smooth the hole using a router, with a bit gliding along the jig. It doesn't matter if the hole is not perfectly round, as long as a pipe will fit into it. It doesn't matter if there is a bit of space here and there between the pipe and the board.

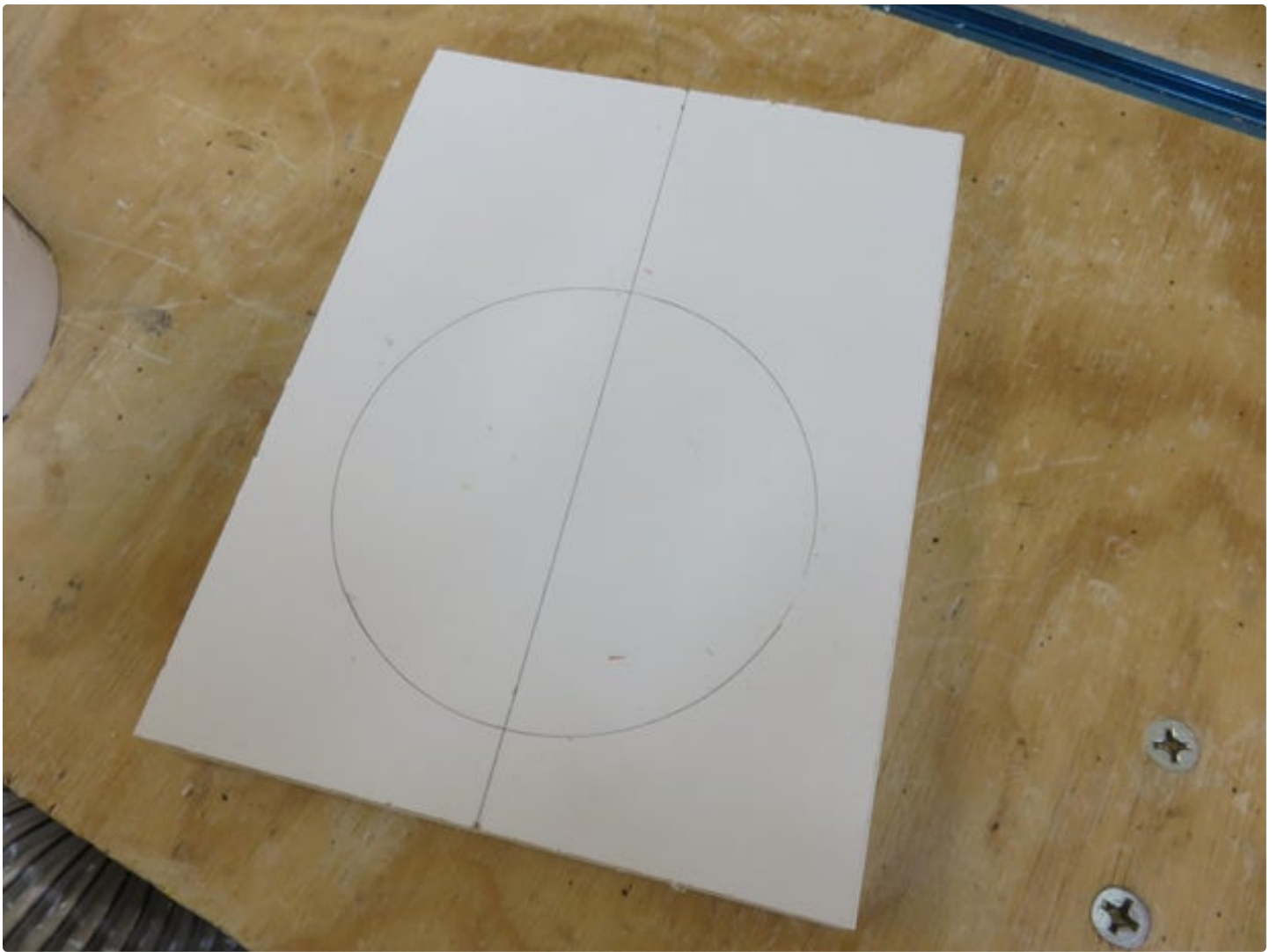
Glue pipe segments into holes in two boards, making sure that the end of the pipe doesn't protrude and that glue doesn't drip into what will be the inside of the gate. You need a strong adhesive for this - I used 3M 4693. Wood glue will not stick to the plastic pipe.

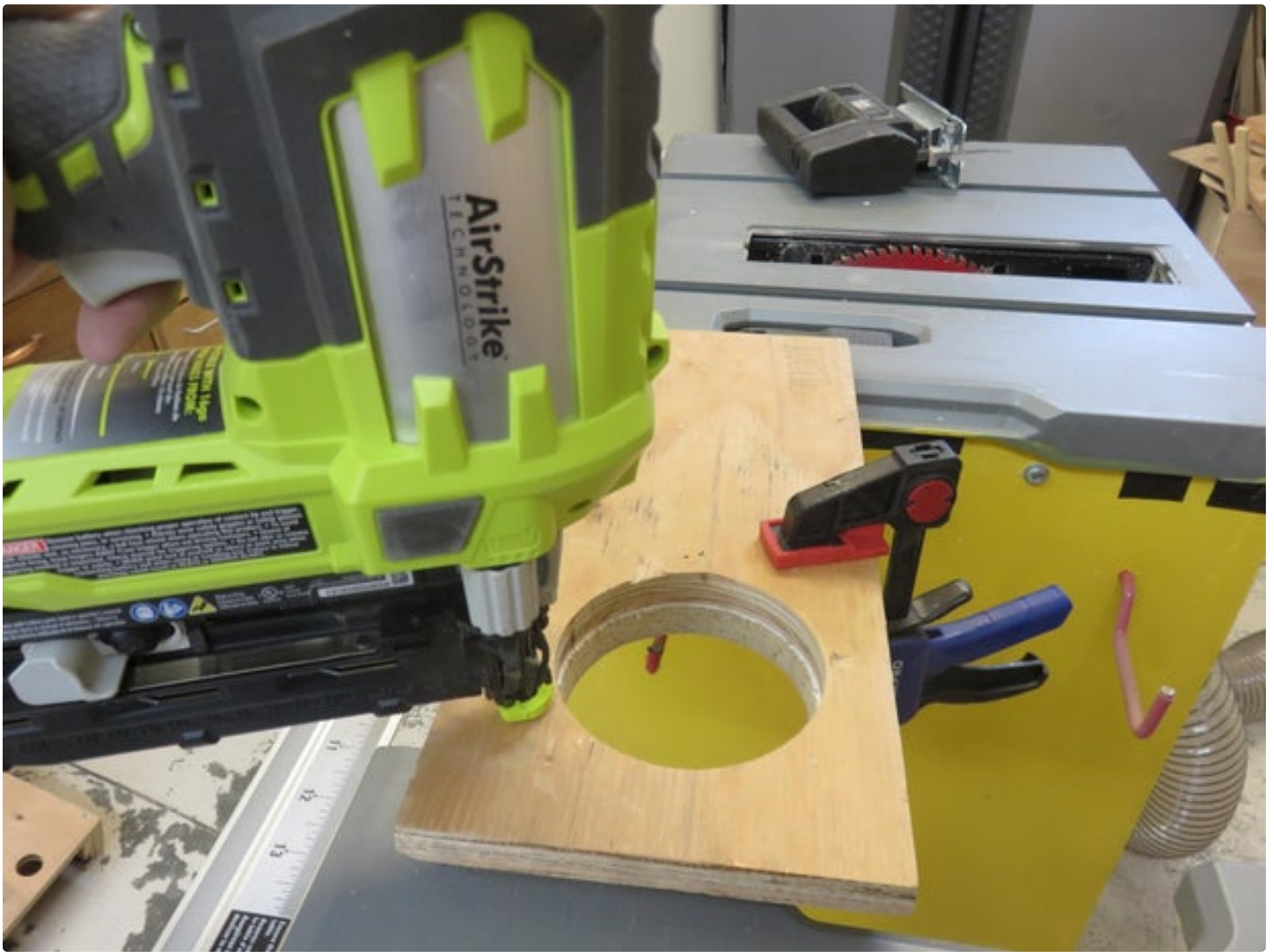
I mostly built the moving gate by gluing a piece of laminate to a piece of HDPE. This is not really needed, HDPE works well just by itself. Once you know the thickness of your gate, make the spacers to be at least a little thicker. I mostly glued two pieces of laminate using contact cement.

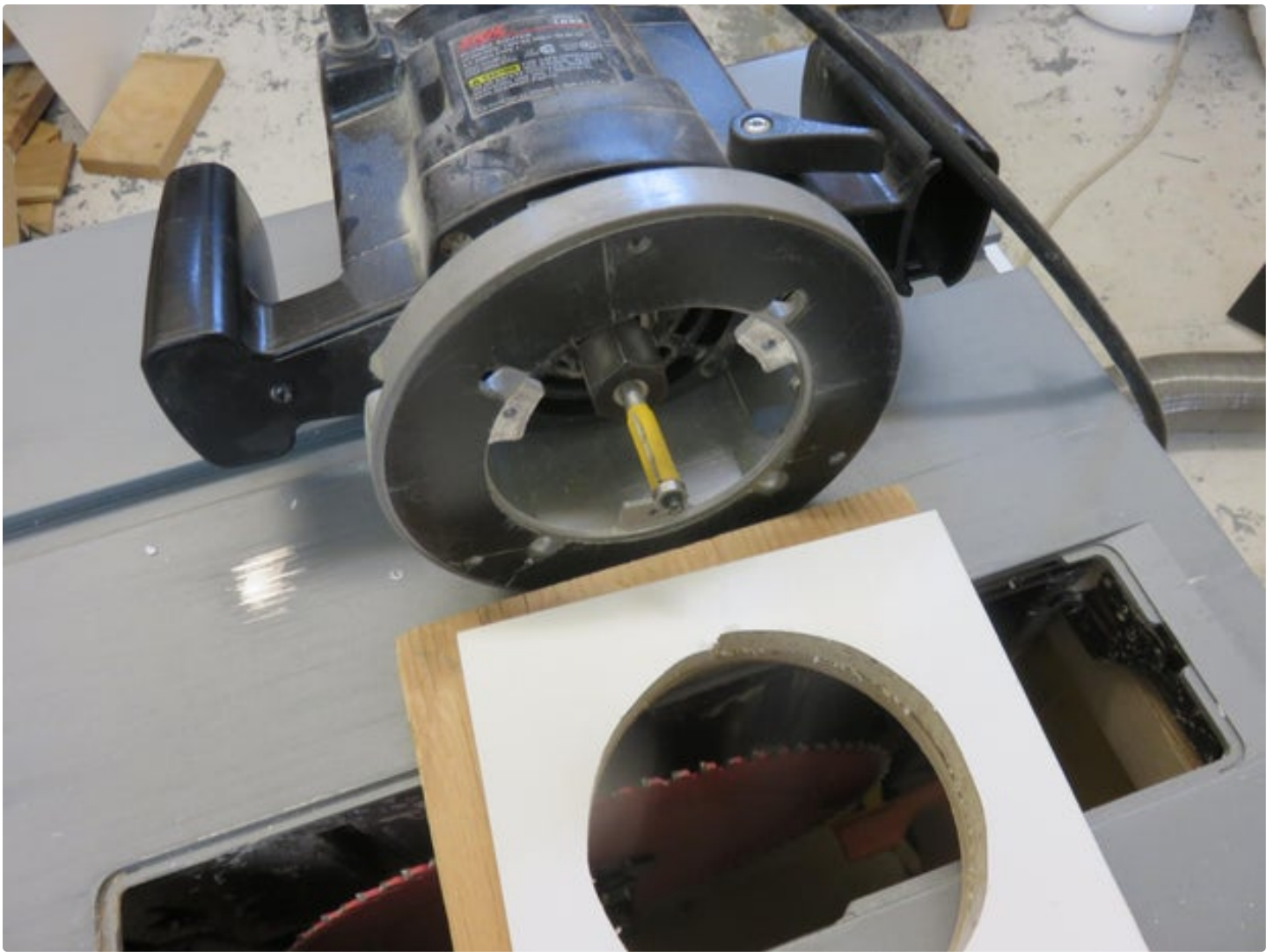
Assemble the gate using the same strong glue. Clamp, let dry and trim anything that protrudes.

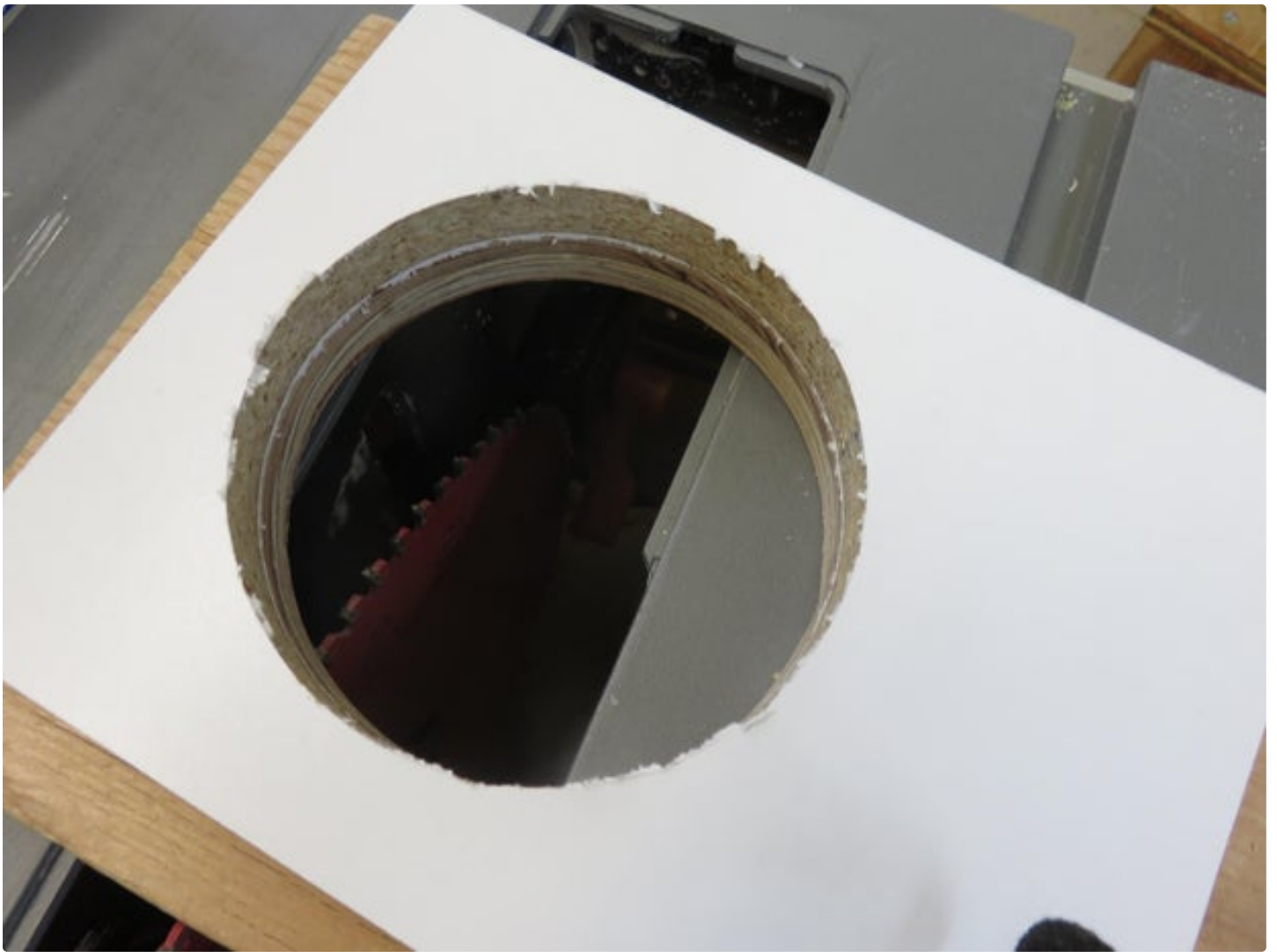




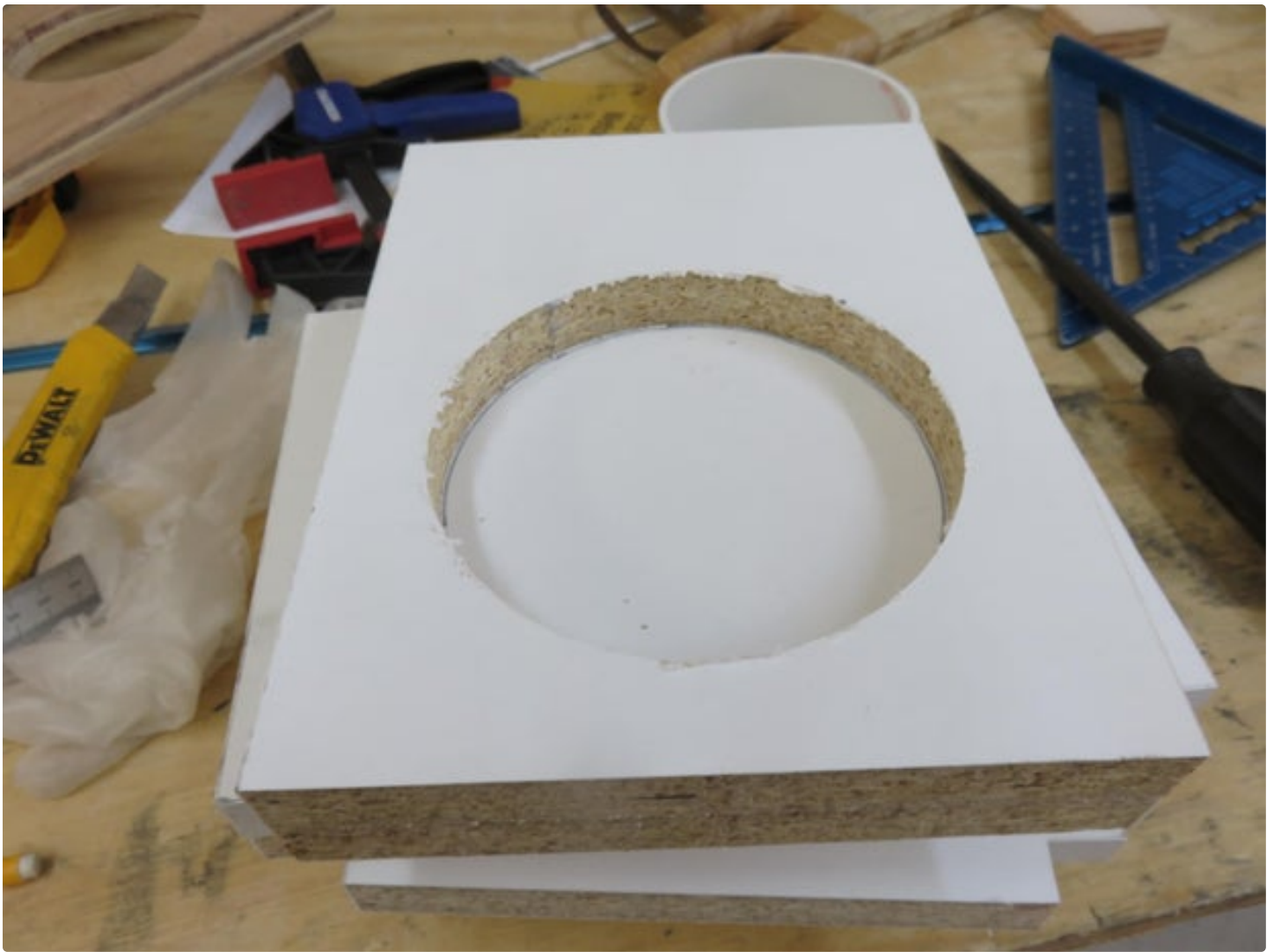




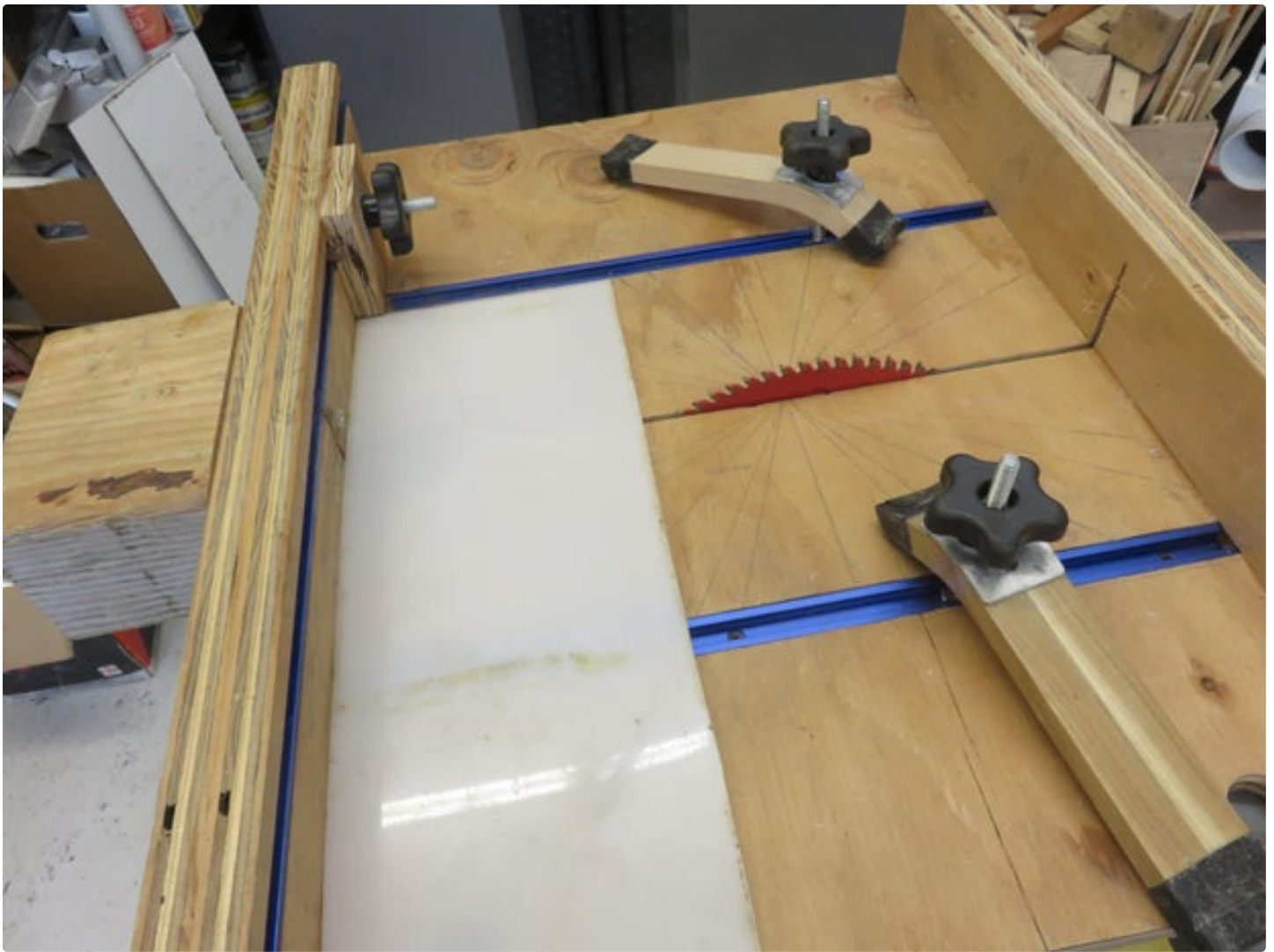






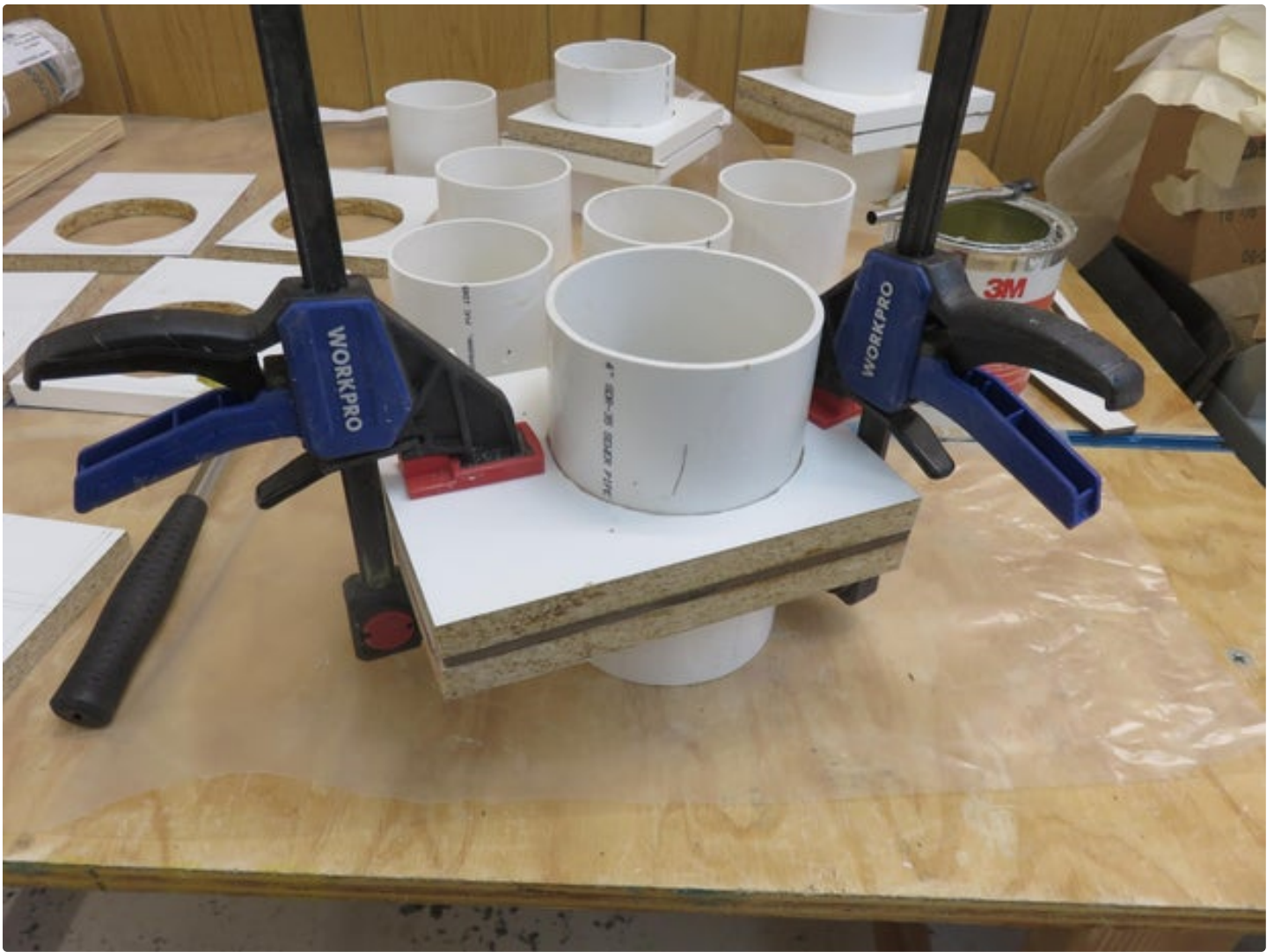


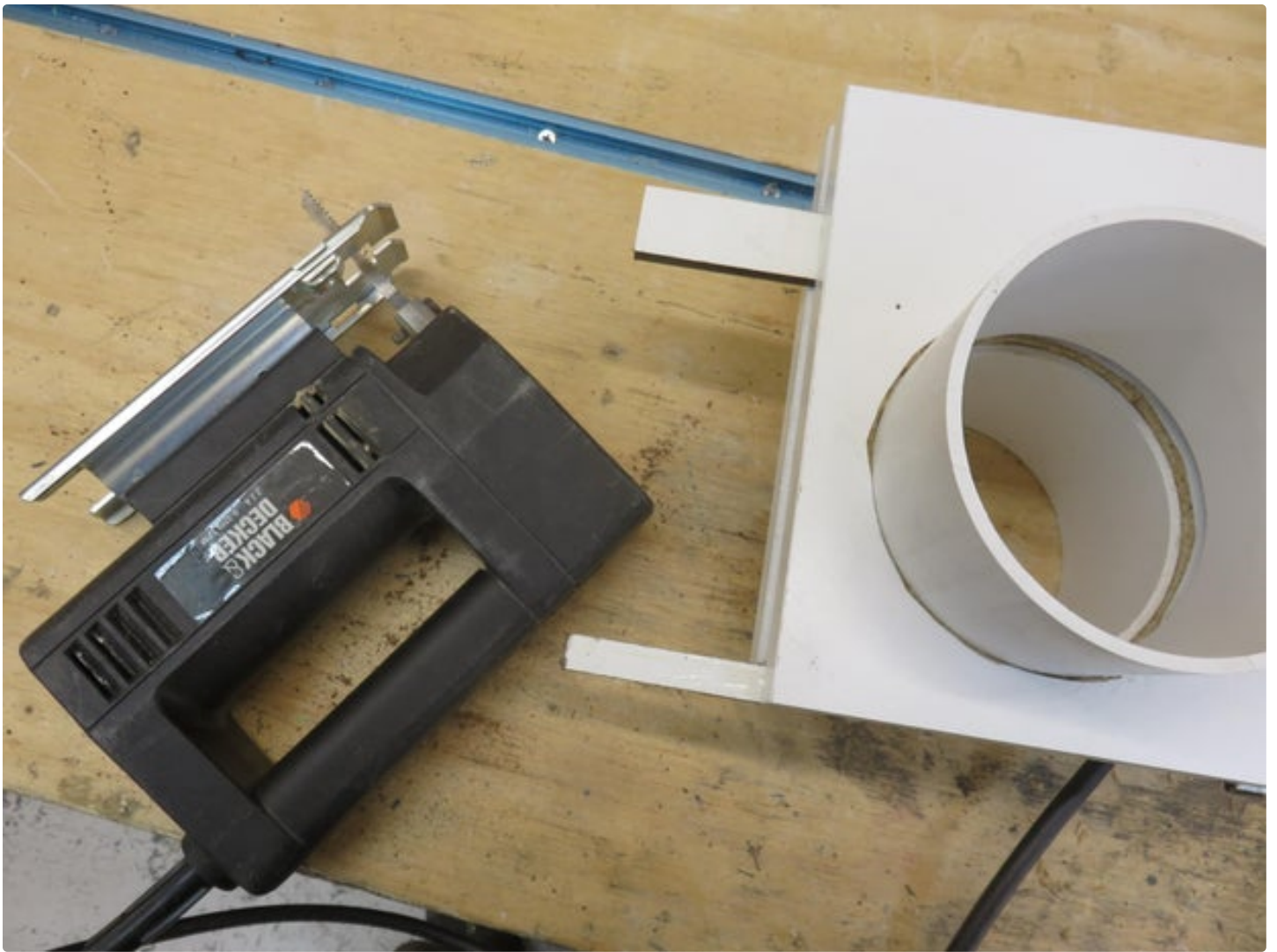














Step 3: Make the Gate Mechanism

I printed the gate mechanism using PLA at 205C, with a heated bed at 60C. Supports are included in the STL. Your slicer may complain about the frame not being a manifold, since the supports are one layer detached from the structure, but I found it sliced fine using slic3r.

To have more secure connection to the motor, I replaced the stock servo connector with a JST-SM female connector.

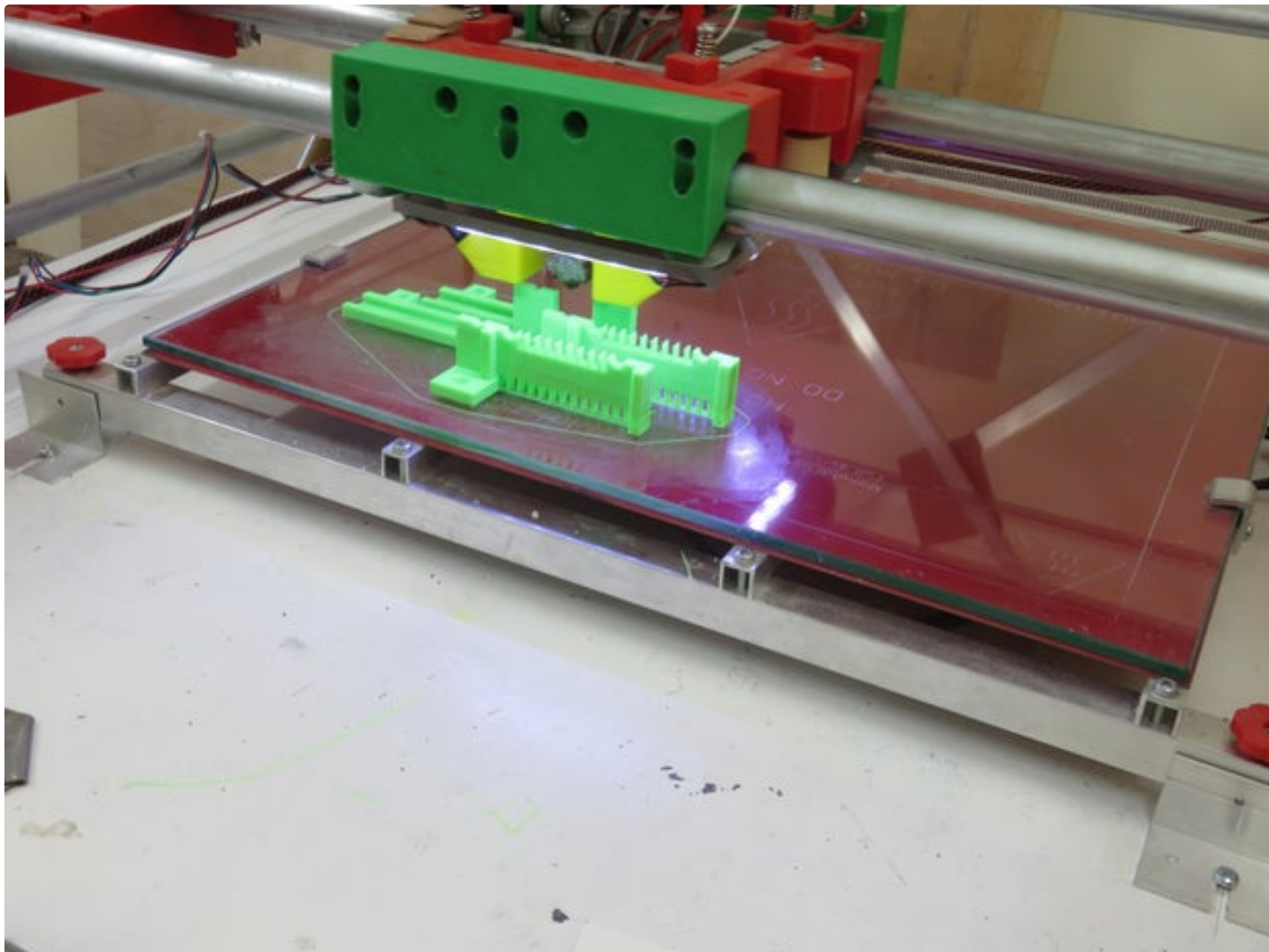
Cut a 65mm or so long piece of 5mm stainless steel rod as an axle. You want to have enough so it protrudes a bit on both sides, since it's used to secure the cover.

We have a frame, two gears, a flat rack gear, two mounting blocks and a cover. The servo attaches to the frame using M3 screws, washers and nuts. There is some freedom to move the motor left or right. Using the gears I printed, the motor needed to be as far right (furthest from the pipe) as possible.

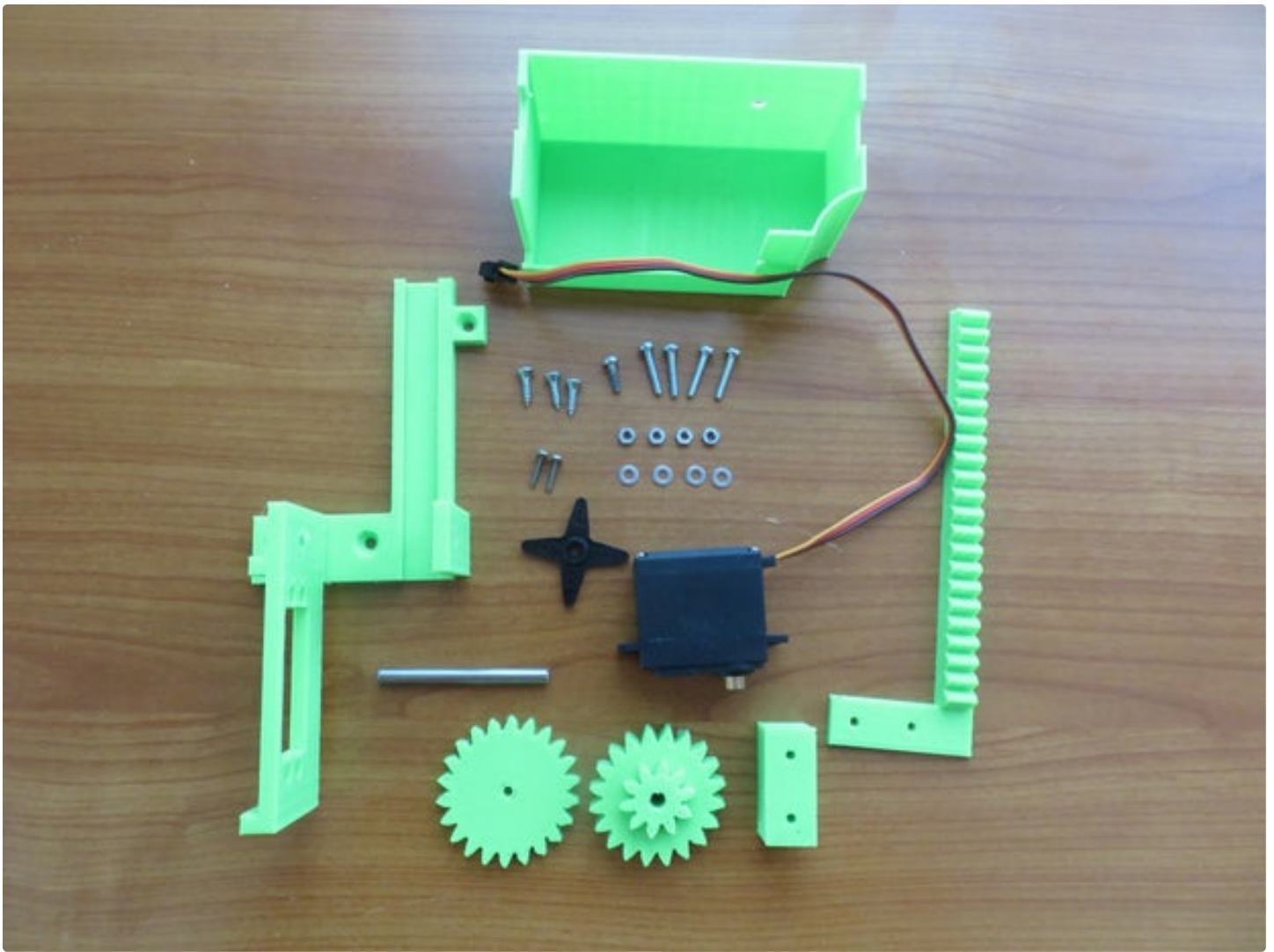
The gear ratios translate 155 degrees of servo rotation into roughly 4" of travel. One gear mounts on the servo, the other on the axle. I cut the screws provided with the servo to length and drilled holes through the gear for two screws to attach the gear to a stock 4 arm servo mount. Use 16mm M3 screw to attach the assembly to the motor.

The 5mm axle should have a snug fit in all four holes - two in the frame, one in the gear and one in the cover. I used a 5mm reamer drill bit to achieve a tight fit. There is enough space so that the gear mounted on the axle can be moved to disengage the motor and adjust the gate. If the holes are loose, the axle may move or the gear may slide on the axle. If

this starts happening, a little piece of electrical tape on the axle between the frame and the gear helps.

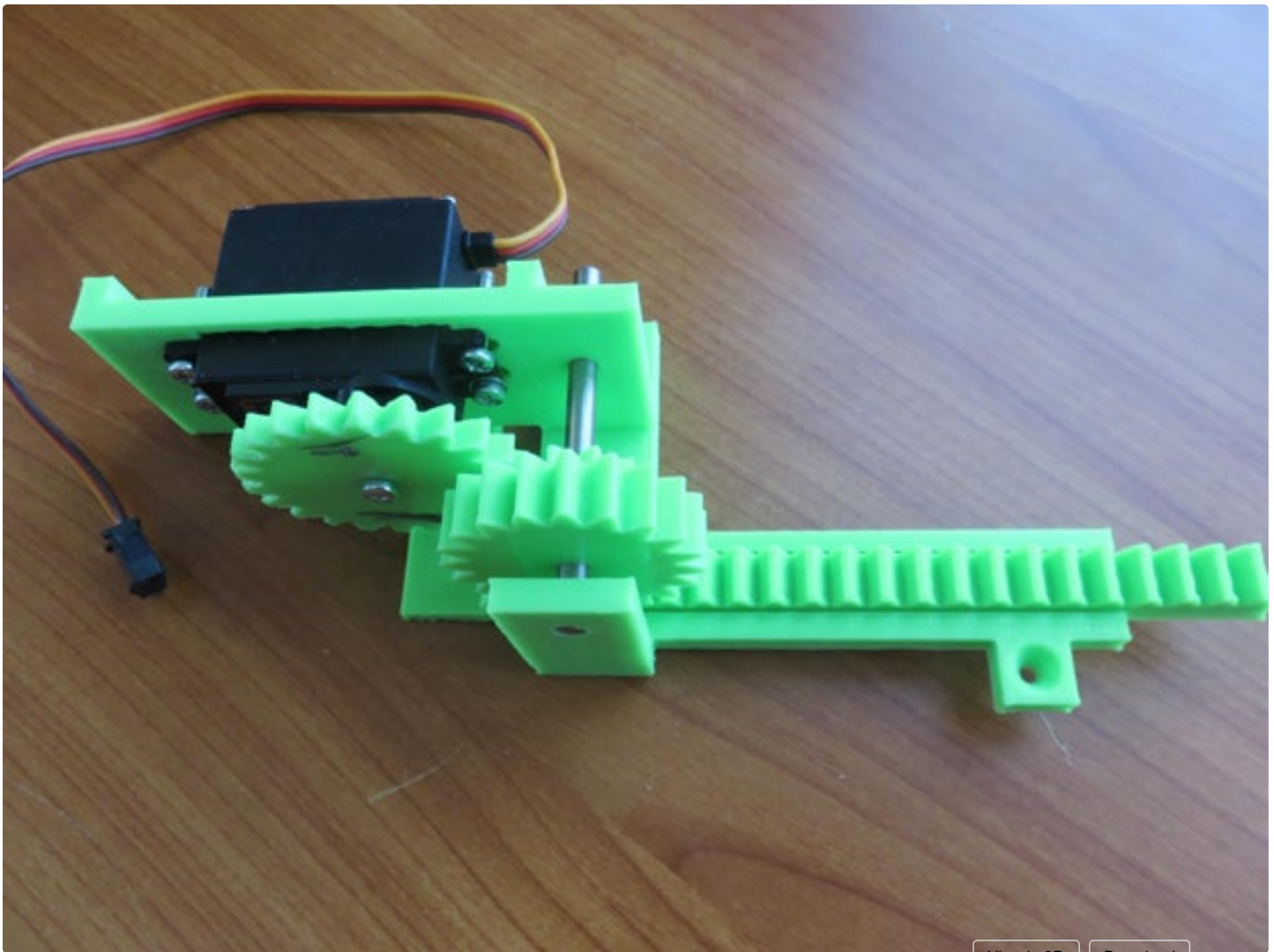












[View in 3D](#)[Download](#) <https://www.instructables.com/ORIG/F9H/D2GY/LA13G8WR/F9HD2GYLA13G8WR.stl>[View in 3D](#)[Download](#) <https://www.instructables.com/ORIG/FW4/NXY0/LA13G8WS/FW4NXY0LA13G8WS.stl>[View in 3D](#)[Download](#) <https://www.instructables.com/ORIG/FI1/SV5D/LA13G8WT/FI1SV5DLA13G8WT.stl>[View in 3D](#)[Download](#) <https://www.instructables.com/ORIG/FEU/SY5T/LA13G8WU/FEUSY5TLA13G8WU.stl>[View in 3D](#)[Download](#) <https://www.instructables.com/ORIG/FX2/S32Y/LA13G8WV/FX2S32YLA13G8WV.stl>[View in 3D](#)[Download](#) <https://www.instructables.com/ORIG/FAH/8PNU/LA13G8WW/FAH8PNULA13G8WW.stl>[View in 3D](#)[Download](#) <https://www.instructables.com/ORIG/FCO/HTC6/LA13G8WX/FCOHTC6LA13G8WX.stl>

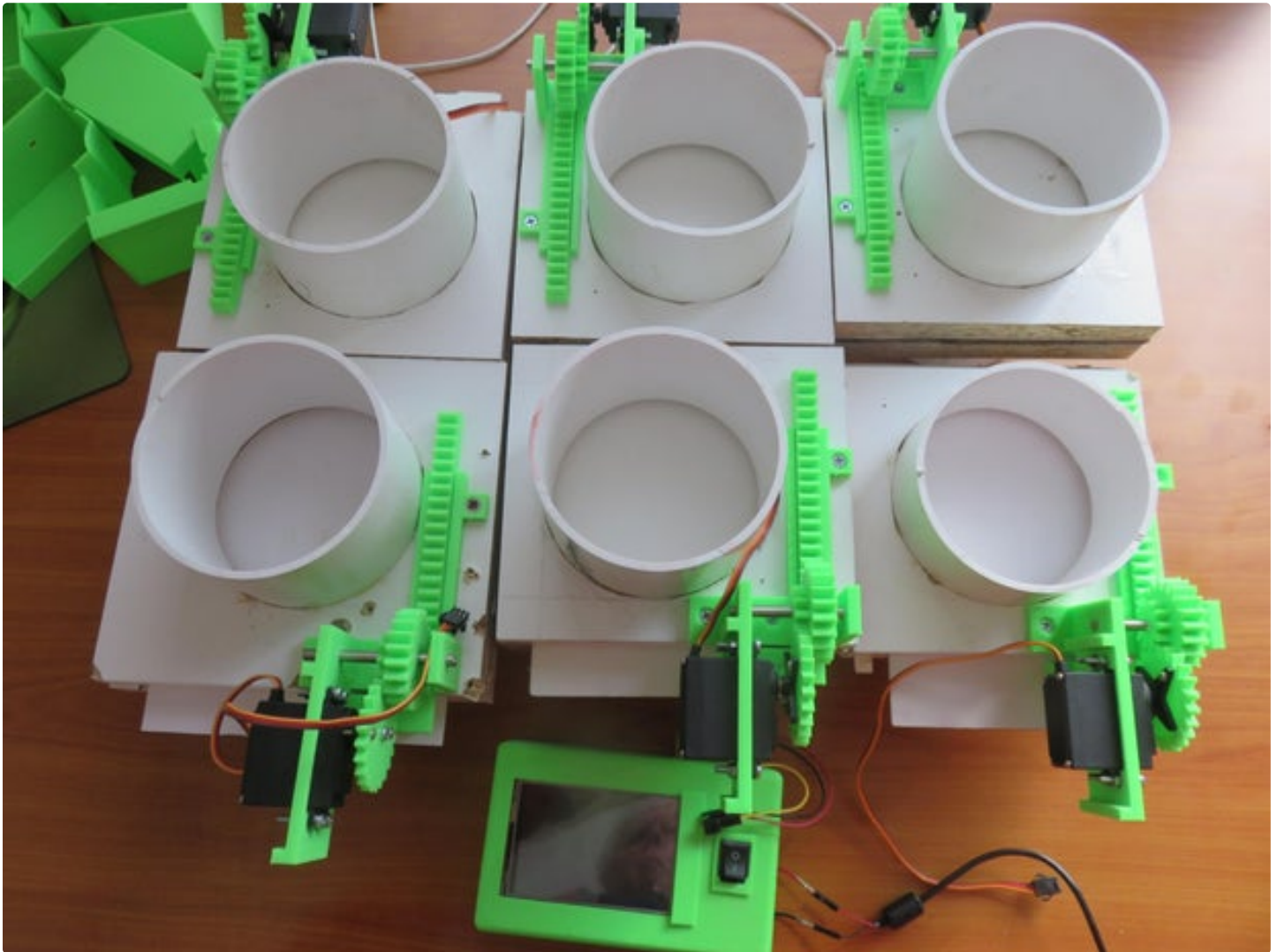
Step 4: Assemble the Gate

Mount the mechanism onto the gate using three 1/2 inch #6 screws. The outer edge of the screw attachments should align with the gate body edge.

The rack has a mounting extension with two screw holes for M3 screws. The thicker spacer block goes between the rack mounting extension and the gate. The thinner spacer block goes on the other side of the gate. Drill holes through the gate to align with mounting holes. They can be slightly oversize so you can adjust the gate position. The screws go through the thinner spacer block, the gate and the thicker spacer block and screw into the rack mounting extension. The holes in the extension are sized so that the threads will hold.

Mounting screws can't protrude too much above the rack mounting extension, or they'll interfere with the frame. You will need a different thicker block and different length of mounting screw if you used different stock to build your gate body.

Try mounting on the mechanism cover - slide the axle towards the motor side until the other end is mostly flush with the frame. The cover should snap in place and the axle can then be pushed back to protrude through the hole in the cover and lock it in place. Remove the cover until the gate is adjusted.

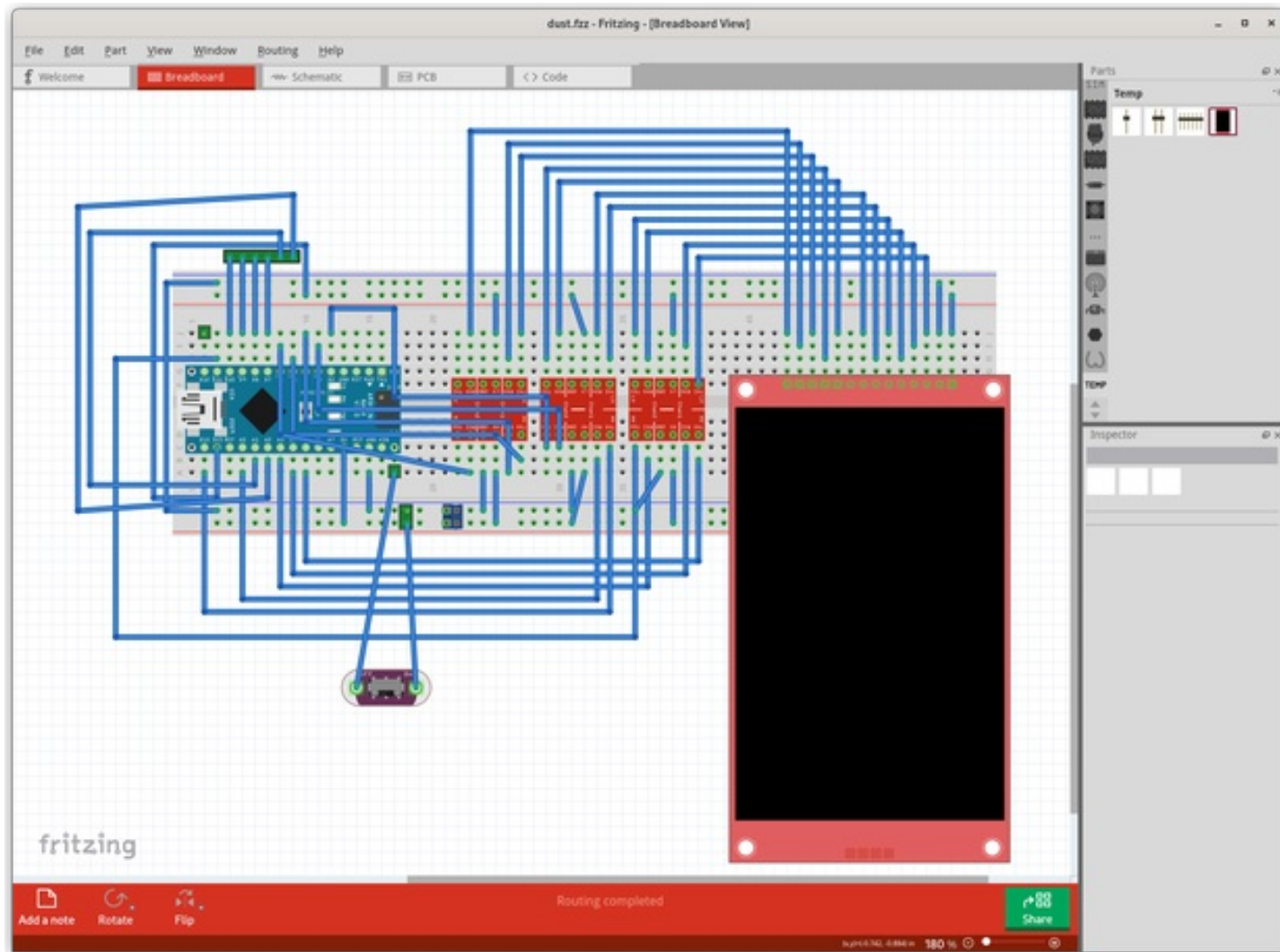


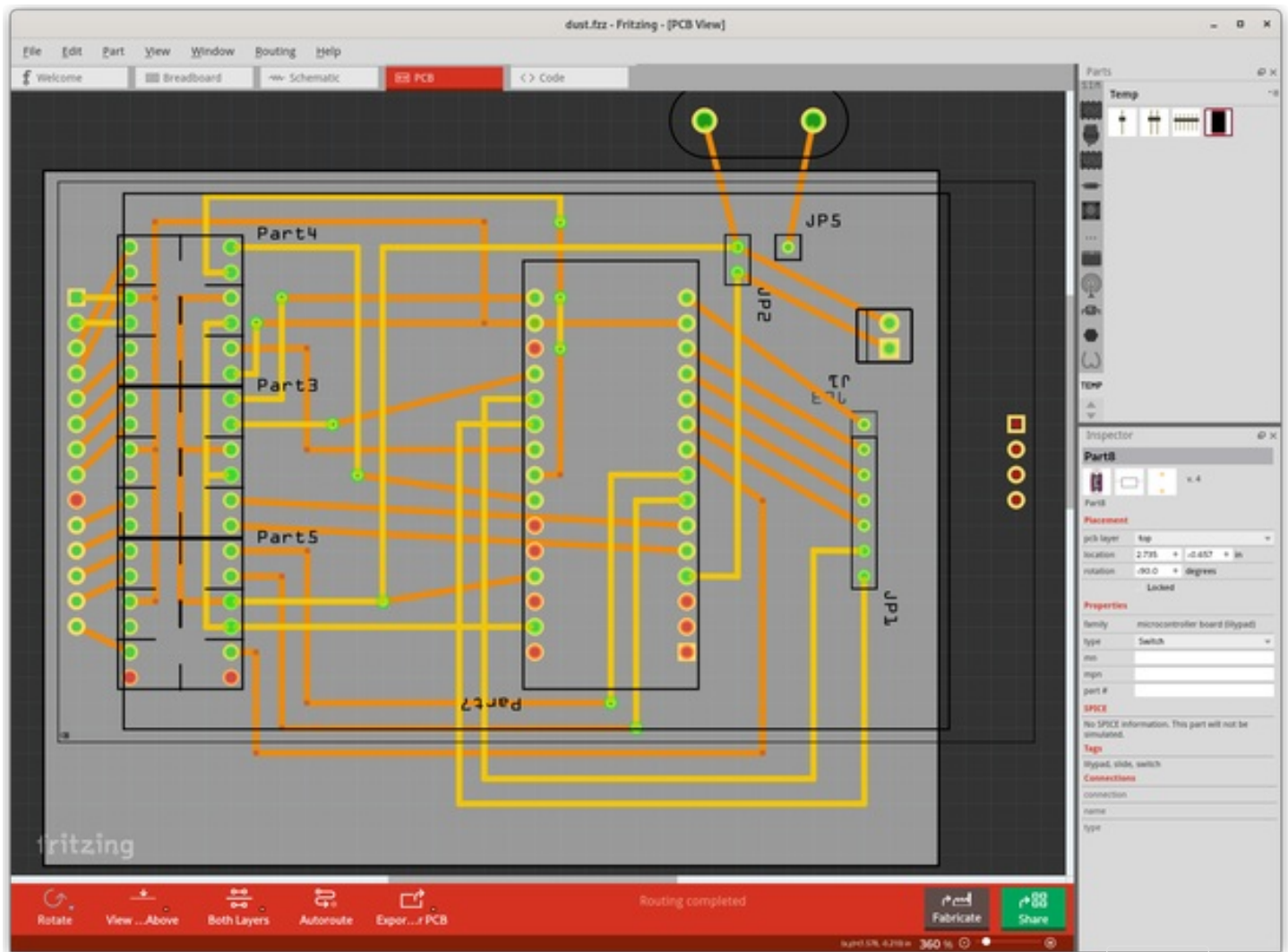
Step 5: Make the Controller

There is a fair amount of soldering to be done here. I enclose the wiring_test.ino sketch that enables you to turn on and off various pins on the Nano and measure with a voltmeter to see that the right part of the circuit is powered.

To make the circuit at least somewhat legible, I used a larger board here than will fit into the controller box. You need to squeeze the circuit onto a 70x50 mm board. I used insulated wrap wire to make most of the connections, so I didn't have to worry about the wires crossing.

The display is not plugged in directly - you need to make a set of three cables (2, 5 and 6 connectors) to connect it to the board. The display is mounted separately on the housing, so the wires should be long enough to support 180 degree twist. Power cables will go into a JST-SM male connector. I am using a reasonably standard 19x12 mm power switch, you may have to trim or shim the switch hole in the housing if you use a different size.





<https://www.instructables.com/ORIG/FA0/C8ZH/LA13GO1X/FA0C8ZHLA13GO1X.fzz>

Download

<https://www.instructables.com/ORIG/F91/3NTT/LA13GQ6R/F913NTTLA13GQ6R.ino>

Download

Step 6: Make and Assemble the Housing

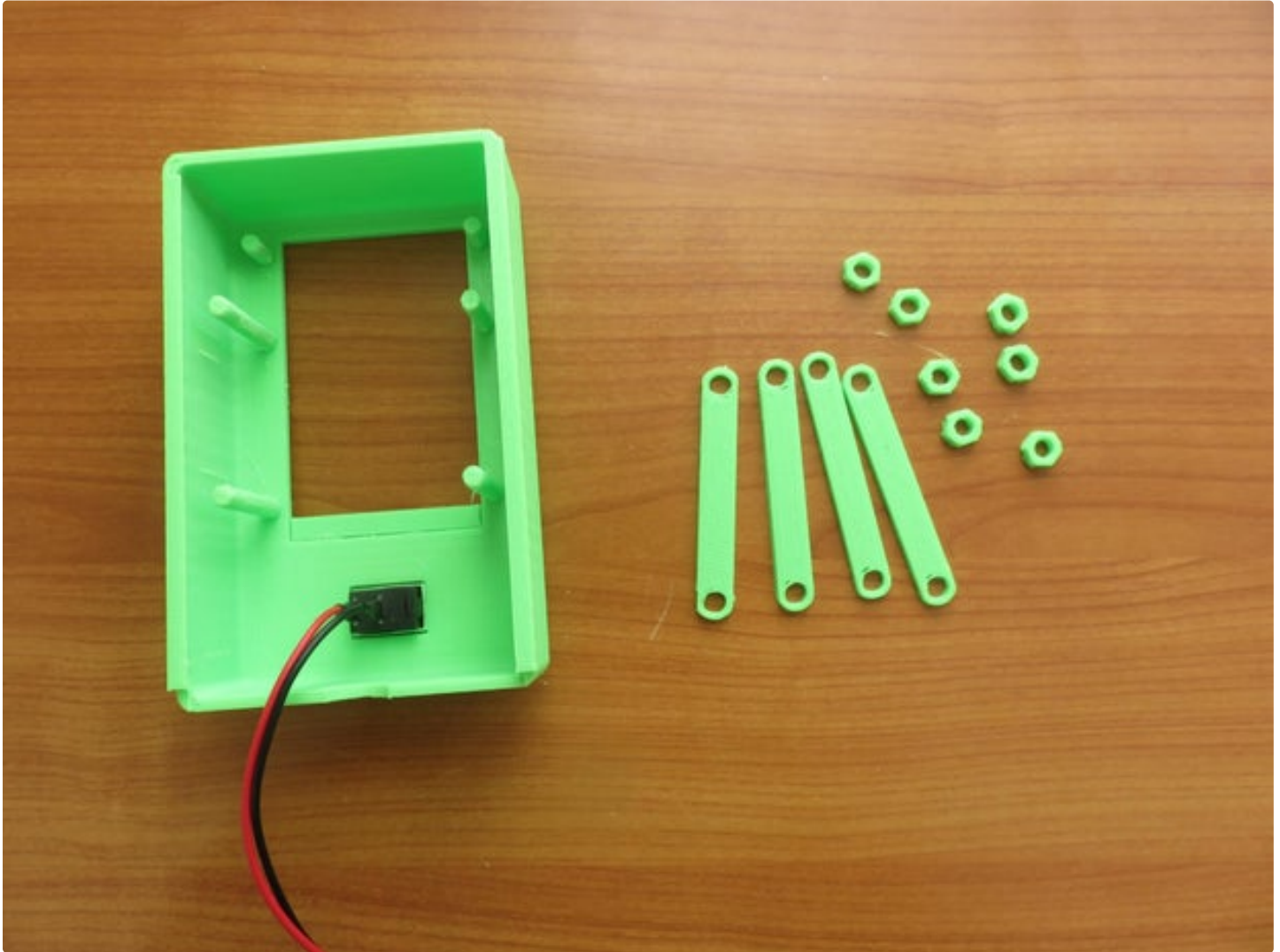
Housing base is designed to be mounted on a wall. Housing cover contains all the electronics and slides onto the base. The cable opening on the base should point down.

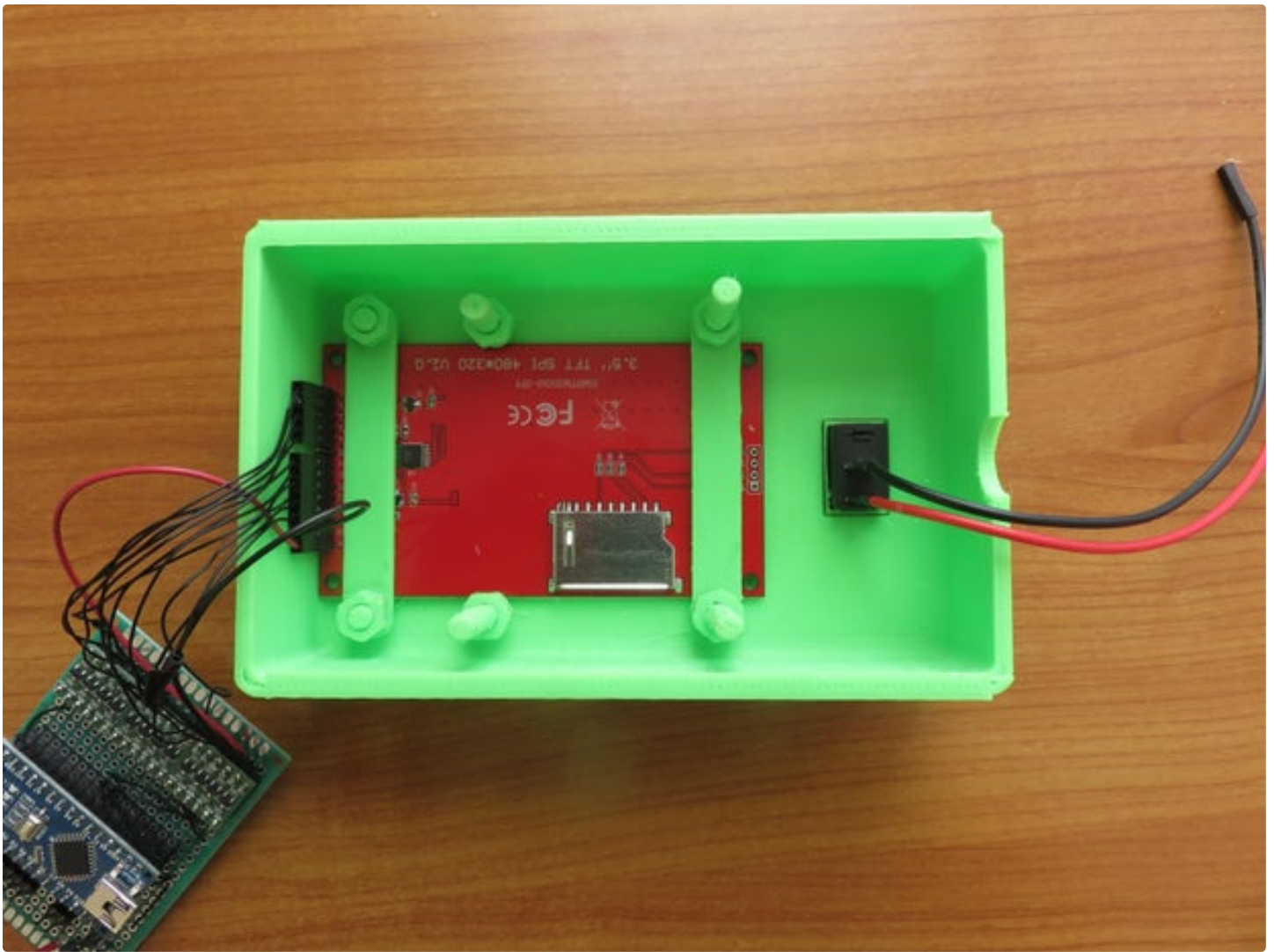
Housing cover has six M6 integrated printed screws used for mounting the controller and the display. You will need 10 M6 printed nuts, 5 thick struts (strut.stl) and one thin strut (strut2.stl). You can use metal M6 nuts if you like (the plastic screws have standard M6-1.0 thread), but it's better to have something non-conductive.

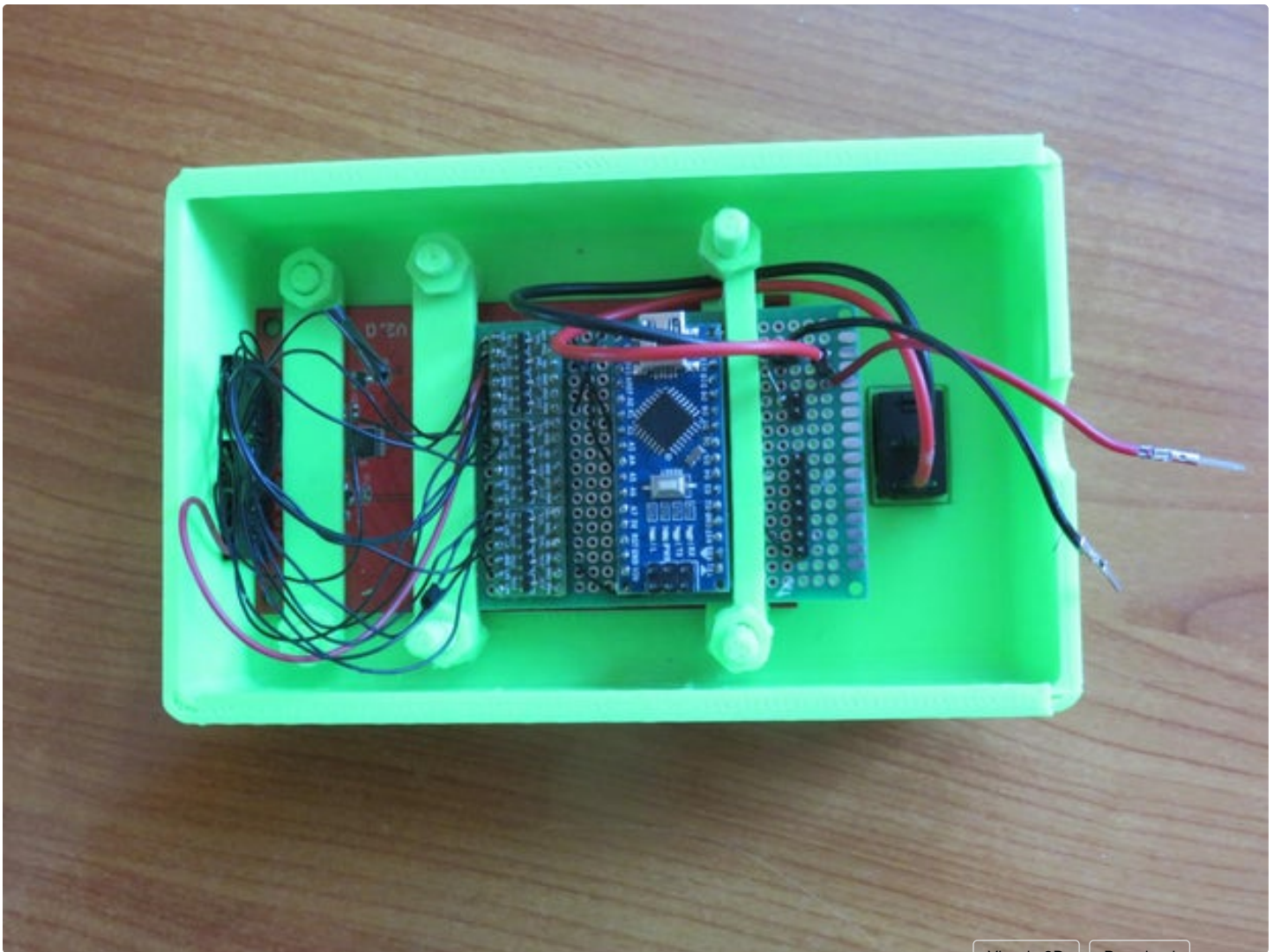
If, like me, you print the housing at too fast a speed, you might have to clean up the integrated M6 screws by carefully threading on a metal M6 nut. Be gentle, one of my screws broke while I was doing this, fortunately leaving enough for the mount.

Slide the display into the opening. Slide two thick struts over the outer sets of screws and tighten them with four nuts. Thread two nuts onto the middle two screws so that they are more or less the same depth as the other screws. Slide two more thick struts over the longer pairs of screws. Place the circuit board onto the struts and slide the two remaining struts, one thick and one thin, over the board. Tighten with the remaining four M6 screws.

The picture I have here is obsolete in the sense that the power takeoff for the servo motors is just a regular connector. This causes all kinds of trouble if you want to use multiple motor circuits, so I replaced it later with a terminal screw block, as is indicated in Fritzing.







	https://www.instructables.com/ORIG/FM1/9BUZ/LA13G8YH/FM19BUZLA13G8YH.stl	View in 3D	Download
	https://www.instructables.com/ORIG/FX9/9O9A/LA13GUQX/FX99O9ALA13GUQX.stl	View in 3D	Download
	https://www.instructables.com/ORIG/FC3/XPUU/LA13GURO/FC3XPUULA13GURO.stl	View in 3D	Download
	https://www.instructables.com/ORIG/FQI/I8J1/LA13GURP/FQII8J1LA13GURP.stl	View in 3D	Download
	https://www.instructables.com/ORIG/F16/P32M/LA13GURQ/F16P32MLA13GURQ.stl	View in 3D	Download

Step 7: The Control Sketch

The main problem in writing a GUI on a small controller is the sad lack of memory. This little package solves the memory problem by keeping most of the object info in program memory. If you look at the code, you will see six objects. Each has a related struct that's kept in memory. Three upper buttons are centered, the lower three are positioned explicitly on the x axis. If you look at the display, if you order the buttons from top to bottom (vac2 goes after vac1), they correspond to the control pins on the board going from bottom to the top.

If you change labels, you need to change the text length in the struct below. Since this is all constant, we can't just use

strlen. The button is dimensioned with one extra space and the label is centered. To modify the interface for your system, you don't need to do anything except modify the labels and perhaps move the buttons around.

Example below shows code for one button. We have three parts. Label and struct contain the fixed data and are kept in PROGMEM. The actual object points to the struct which, in turn, points to the label. Note that we have two color pairs. BLUE and YELLOW are "off" foreground and background, while WHITE and RED are "on" foreground and background.

The label is 6 characters long, while we pass 7 to the STR_PIXW macro.

The actual sketch uses the "clickable" object, PROGMEM data is accessed using relevant object methods.

```
const char PROGMEM label_sander_button[] = "Sander" ;
const struct clickable_prog PROGMEM buff_sander_button =
{
  (C_LCD_W-STR_PIXW(7,4))/2, 30, STR_PIXW(7,4), 70, // uint16_t
  BLUE, YELLOW, // uint16_t
  false, // bool
  4, // uint8_t
  // if any of the strings are missing, set to NULL
  (__FlashStringHelper *)label_sander_button,
  NULL, NULL,
  6, // uint8_t
  WHITE, // uint16_t
  gate_action,
  (long *)0, // no need for ptr, index into EEPROM and servos
  RED // uint16_t bg on color
};

clickable sander_button( &buff_sander_button, AM_ONOFF | CHG_ON );
```

See the [touchscreen](#) Instructable if you want to learn how it all works.

While a motor is moving, the screen doesn't react to input.

The controller keeps the gate status in EEPROM. It will move the gates to the remembered state on power on.



<https://www.instructables.com/ORIG/FJ4/792D/LA13H45U/FJ4792DLA13H45U.ino>

Download



<https://www.instructables.com/ORIG/FJK/V1OF/LA13H45V/FJKV1OFLA13H45V.cpp>

Download



<https://www.instructables.com/ORIG/FP3/7D1O/LA13H45W/FP37D1OLA13H45W.h>

Download

Step 8: Assemble the System

Cut and connect your piping depending on your needs. My system uses 45 and 90 degree elbows, 45 and 90 degree junctions, as well as pipe connectors. A standard 4" OD dust fittings will fit into the 4" ID sewer pipe. Most of the gates in my system move horizontally, but a couple move at a 45 degree angle.

For gates close to the controller, you can use a phone wire. For longer runs, you'll need a thicker cable to run power to the motors, phone wire has too much resistance. Phone wire is fine for the signal line. I used JST-SM connectors throughout.

To calibrate a gate, move the gear on the axle so it's disconnected from the motor. Open the gate on the controller. Move the gate so the tip of the gate is barely within the pipe and move the axle gear back so it engages with the motor. The gate should now open and close properly.

Finish by mounting the cover.

I move my table saw, so it's connected to the system via a 4" flexible pipe. You can see the mechanism with and without the cover. Like in every workshop, the space is at premium in mine, so I mounted the dust collection bag above the blower. You can see the sander, the fan on/off switch (it will run just barely on a 20 amp breaker, but I have a 30amp circuit connected here), the controller and the dust collector. Three of my gates are accessible, but three are tucked away below the workbench.

I'm pretty happy how it all works - so much air is flowing that my miter box, a perennial source of dust, needs only a simple hood.

