```
//Steampunk style microcontroller controlled physical music instrument.



//Organ pipe part:

SPEED_OF_SOUND_IN_AIR_20C = 343.2;

function waveLengthInmmFromFreq(frequency, closed) = (1000.0 * SPEED_OF_SOUND_IN_AIR_20C / frequency) * (closed ? 0.5 : 1.0);


module organPipe(frequency = 880, diameter=50, closed=true, fanSide=80, fanThickness=20) {
  wavelength = waveLengthInmmFromFreq(frequency, closed);
  sideWall = 3;
  endWall = 5;
  radius = diameter * 0.5;
  startLen = 50;
  windPipeRadius = 10;
  windPipeLen = 10;
  fanRadius = fanSide * 0.5;
  fanAdapterLen = fanRadius;

  difference() {
    union() {
      // Pipe
      translate([0,0,fanThickness + fanAdapterLen + windPipeLen + startLen])
        cylinder(h = wavelength + endWall, r = radius + sideWall);

      // Start section
      translate([0,0,fanThickness + fanAdapterLen + windPipeLen])
        cylinder(h = startLen, r1 = windPipeRadius + sideWall, r2 = radius + sideWall);

      // Wind section
      translate([0,0,fanThickness + fanAdapterLen])
        cylinder(h = windPipeLen, r1 = windPipeRadius + sideWall, r2 = windPipeRadius + sideWall);

      // Fan adapter
      translate([0,0,fanThickness])
        cylinder(h = fanAdapterLen, r1 = fanRadius + sideWall, r2 = windPipeRadius + sideWall);
    }

    // Pipe cavity
    translate([0,0,fanThickness + fanAdapterLen + windPipeLen + startLen])
      cylinder(h = wavelength, r = radius);

    // Start section cavity
    translate([0,0,fanThickness + fanAdapterLen + windPipeLen])
      cylinder(h = startLen, r1 = windPipeRadius, r2 = radius);

    // Wind section
    translate([0,0,fanThickness + fanAdapterLen])
      cylinder(h = windPipeLen, r1 = windPipeRadius, r2 = windPipeRadius);
```

```
    // Fan adapter
    translate([0,0,fanThickness - 1])
      cylinder(h = fanAdapterLen + 1, r1 = fanRadius, r2 = windPipeRadius);
  }

}


organPipe(440*4, 50, true);
```