

Learn to Solder

with ArduTouch Music Synthesizer kit

and make music, sound, and noise!

Mitch Altman

Chief Scientist, **Cornfield Electronics**, San Francisco, CA

Inventor of **TV-B-Gone** universal remote controls

Co-founder of **3Ware** (successful Silicon Valley startup)

Pioneer of **VR** (in the mid-1980s)

Founding mentor at **HAX** (1st and biggest hardware accelerator)

Co-founder of **Noisebridge** (San Francisco hackerspace)

email: mitch@CornfieldElectronics.com

site: www.CornfieldElectronics.com

facebook: maltman23

flickr: maltman23

WeChat: mitchaltman

Fediverse: [@maltman23@mastodon.social](https://maltman23@mastodon.social)

Patreon: mitchaltman

fablab|nk

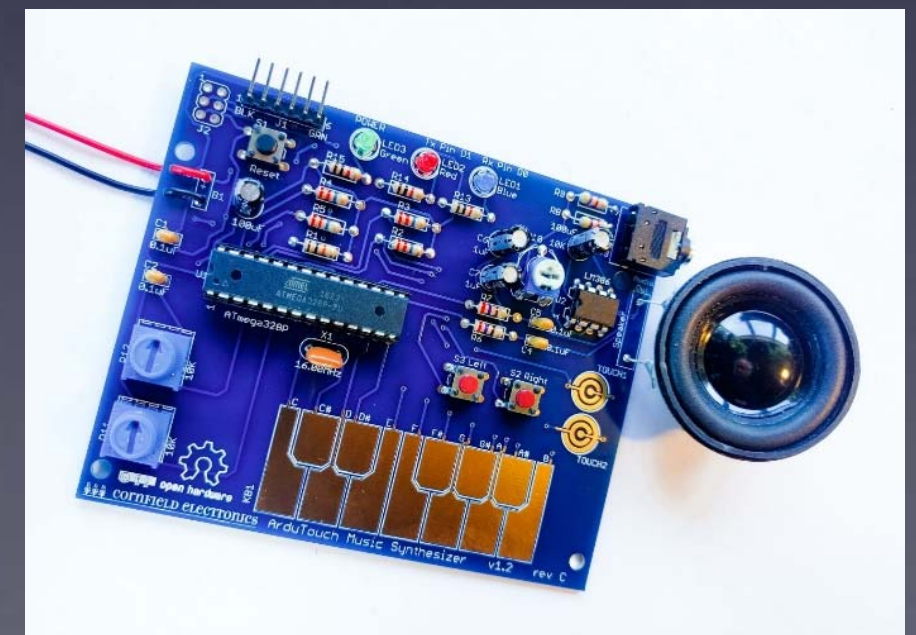


CORNFIELD ELECTRONICS

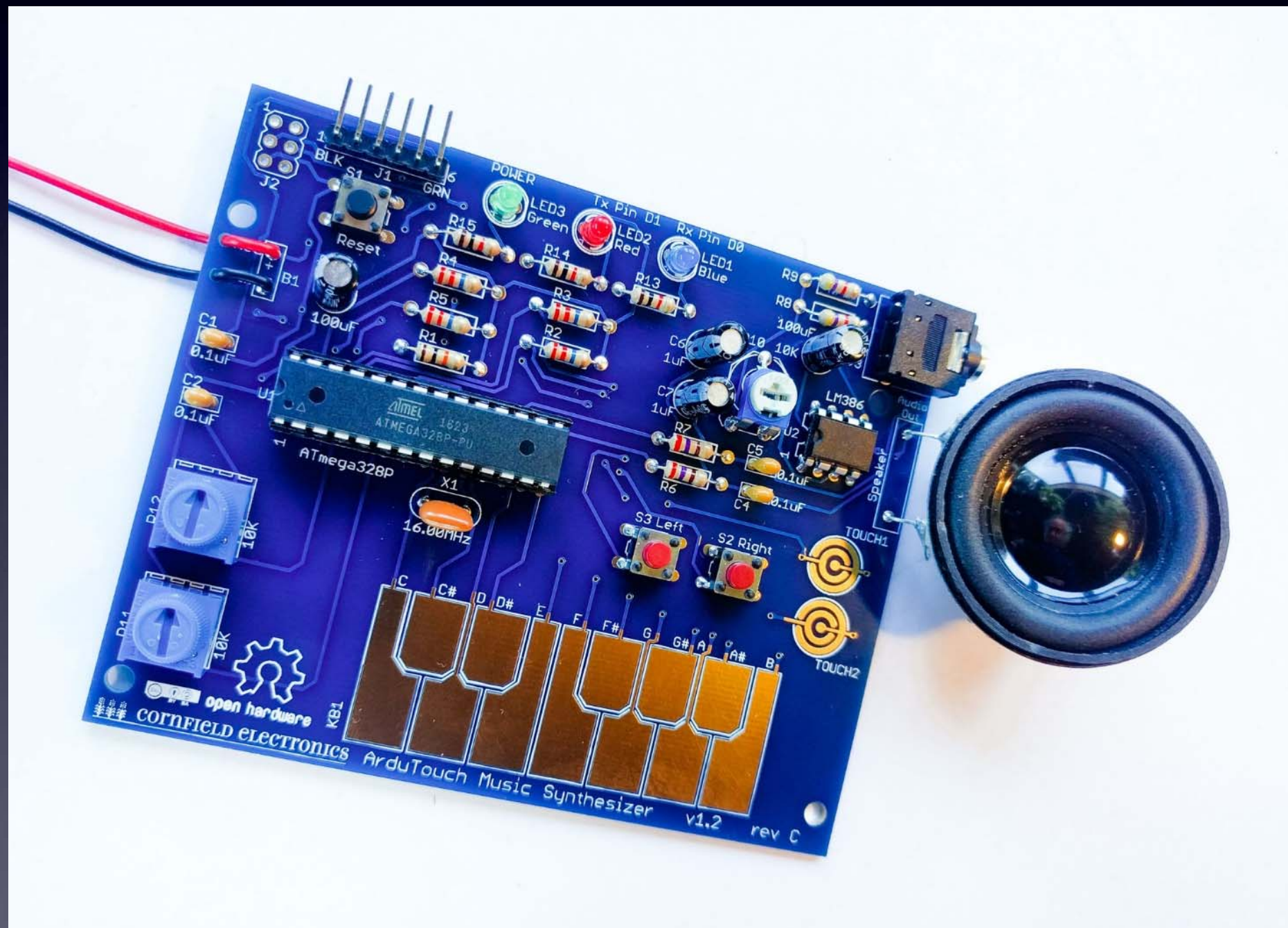
Syllabus

- Intro to ArduTouch music synthesizer kit
- Live demo of ArduTouch
- Intro to music synthesis / Digital Signal Processing
- How to solder
- How to program ArduTouch with Arduino software

Soldering Workshops / kits

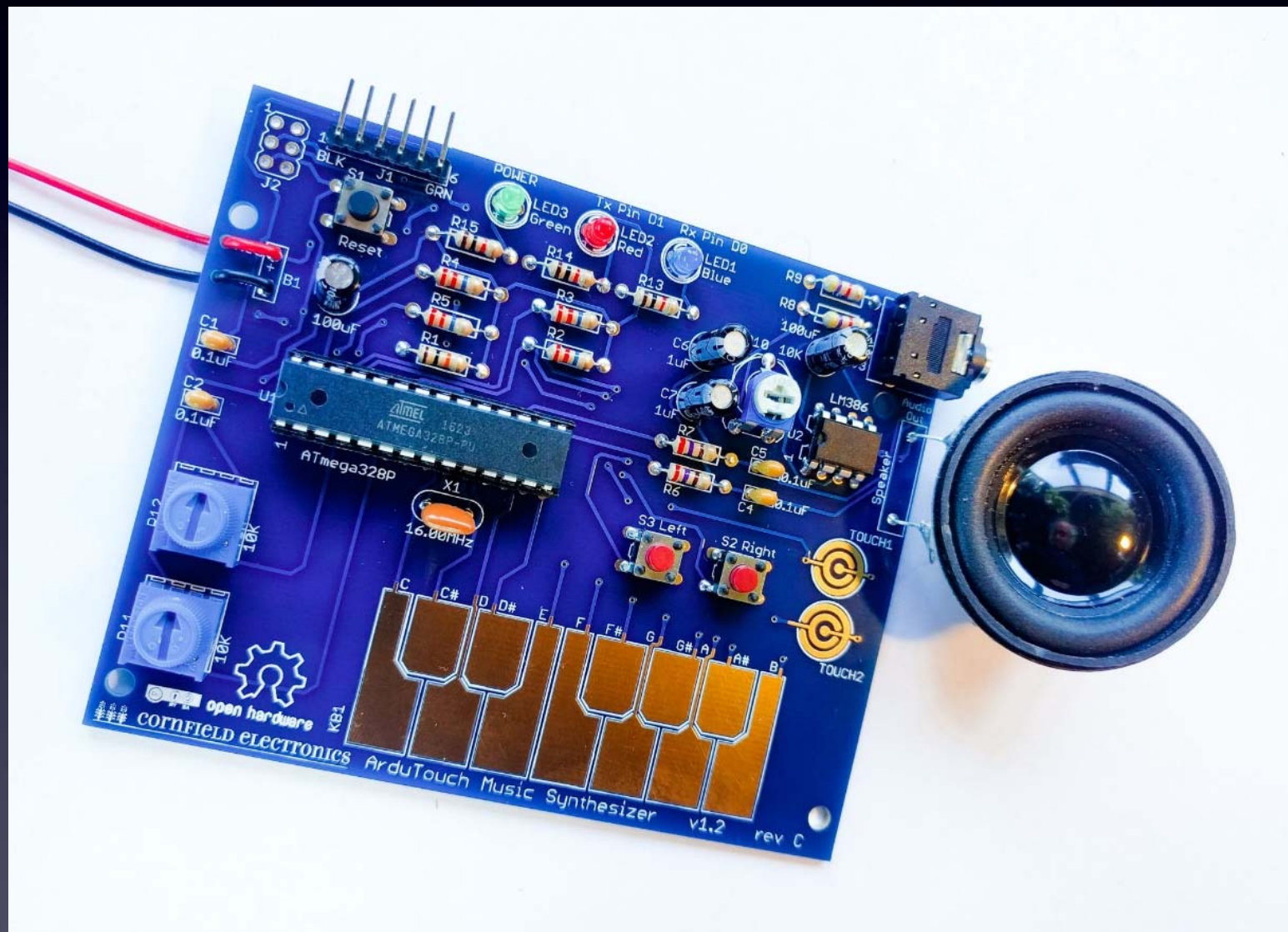


ArduTouch Music Synthesizer



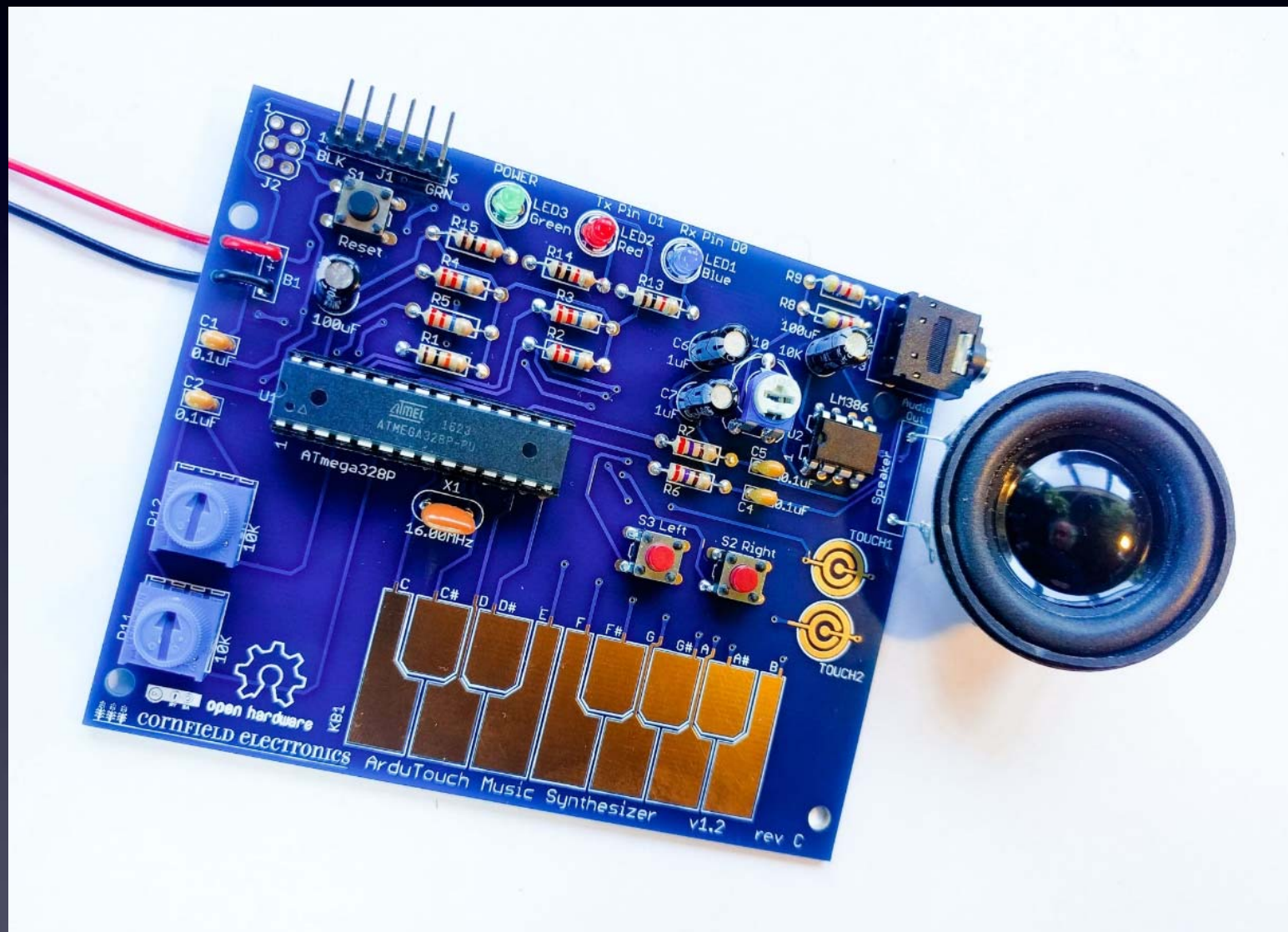
rev C

ArduTouch

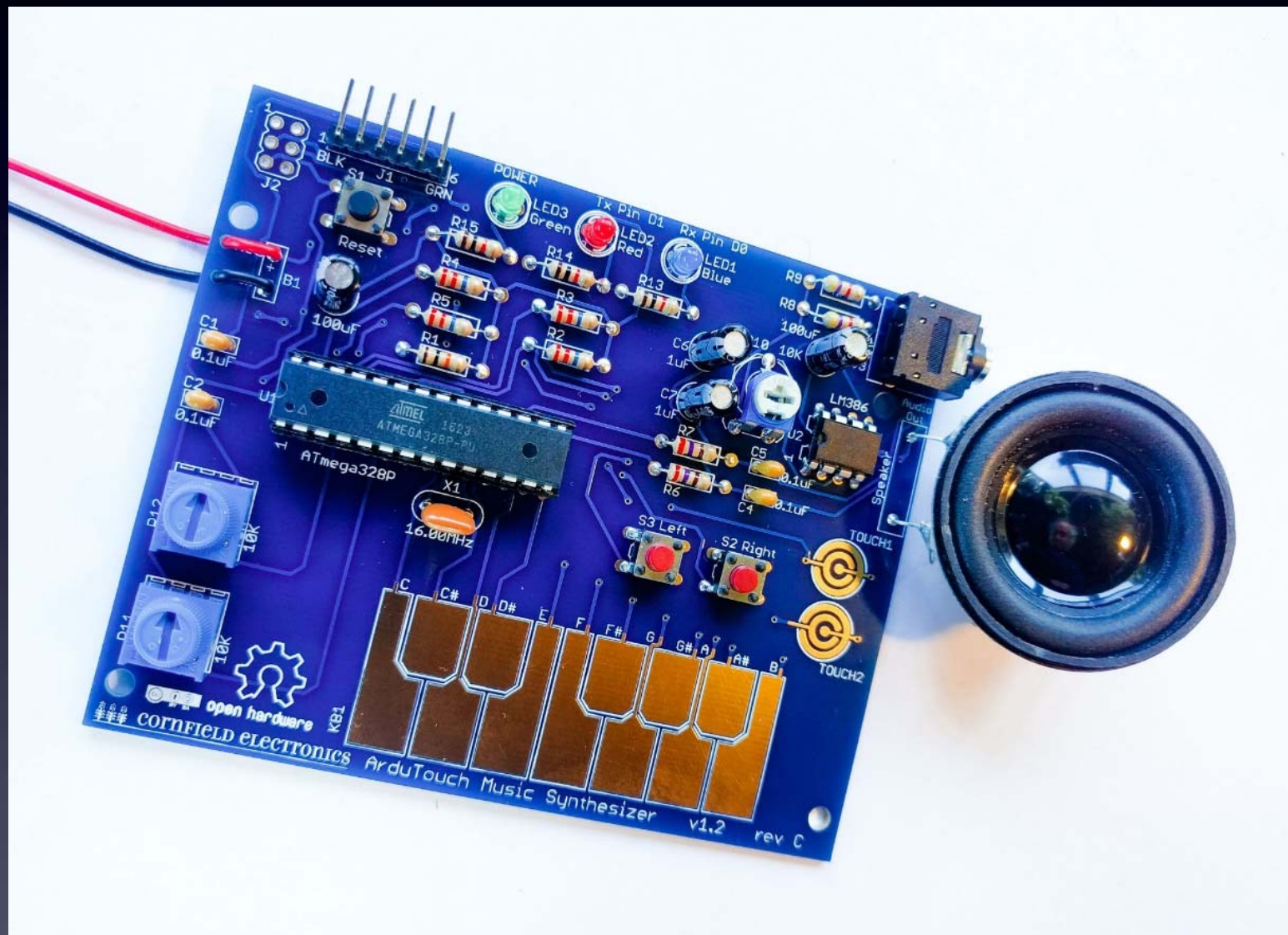


ArduTouch

Great for
learning
to solder



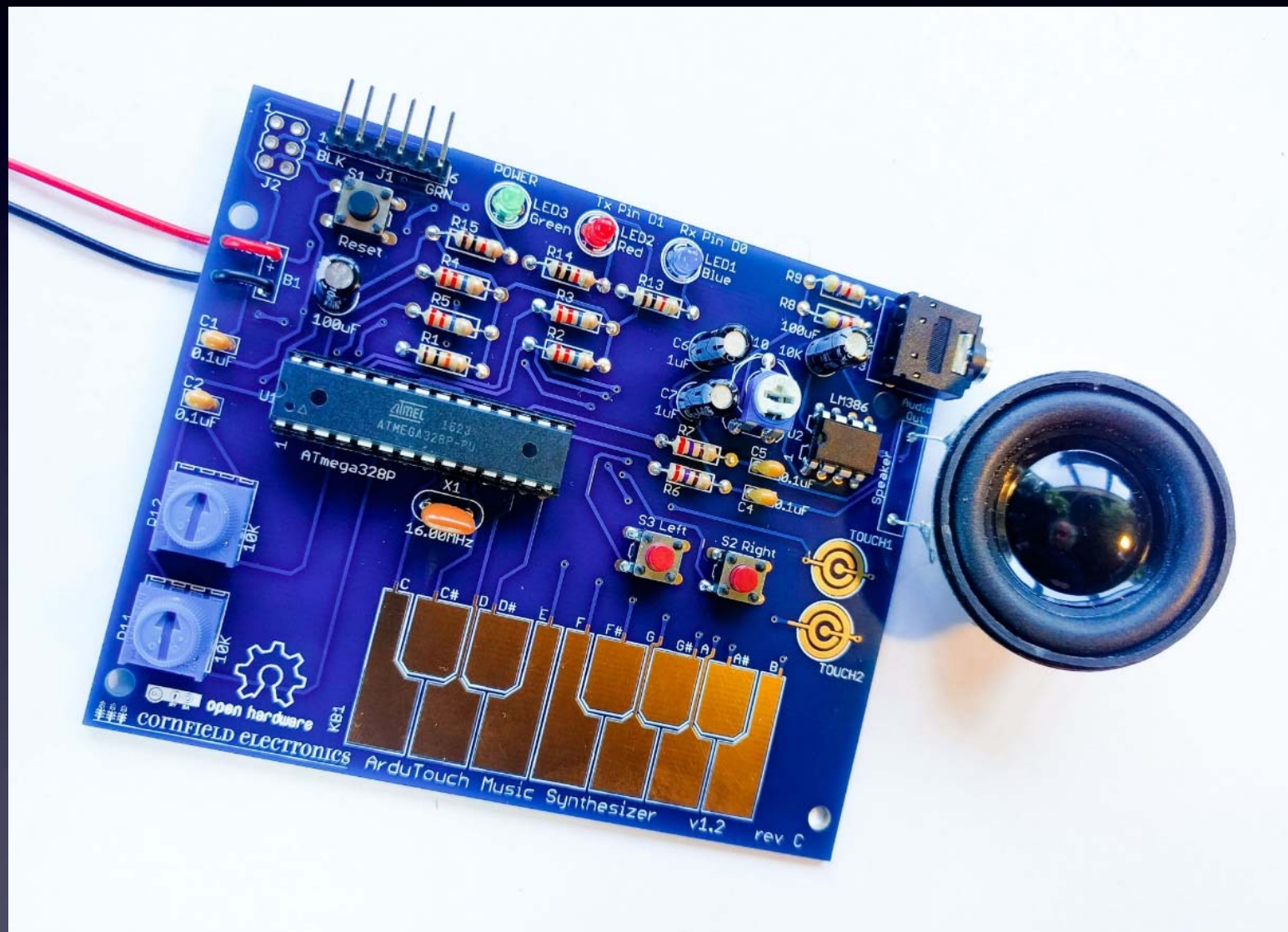
ArduTouch



Solder it together
– and it works!

And you can also
program
your own synthesizers

ArduTouch

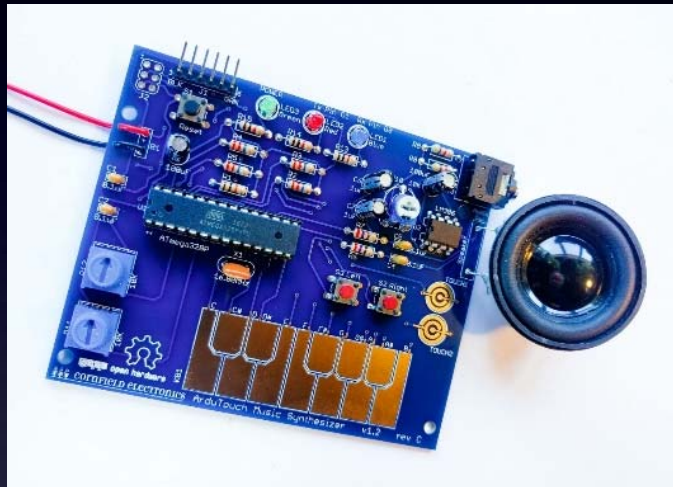


Solder it together
– and it works!

And you can also
program
your own synthesizers

You can also
learn
Digital Signal Processing

ArduTouch



Live demo

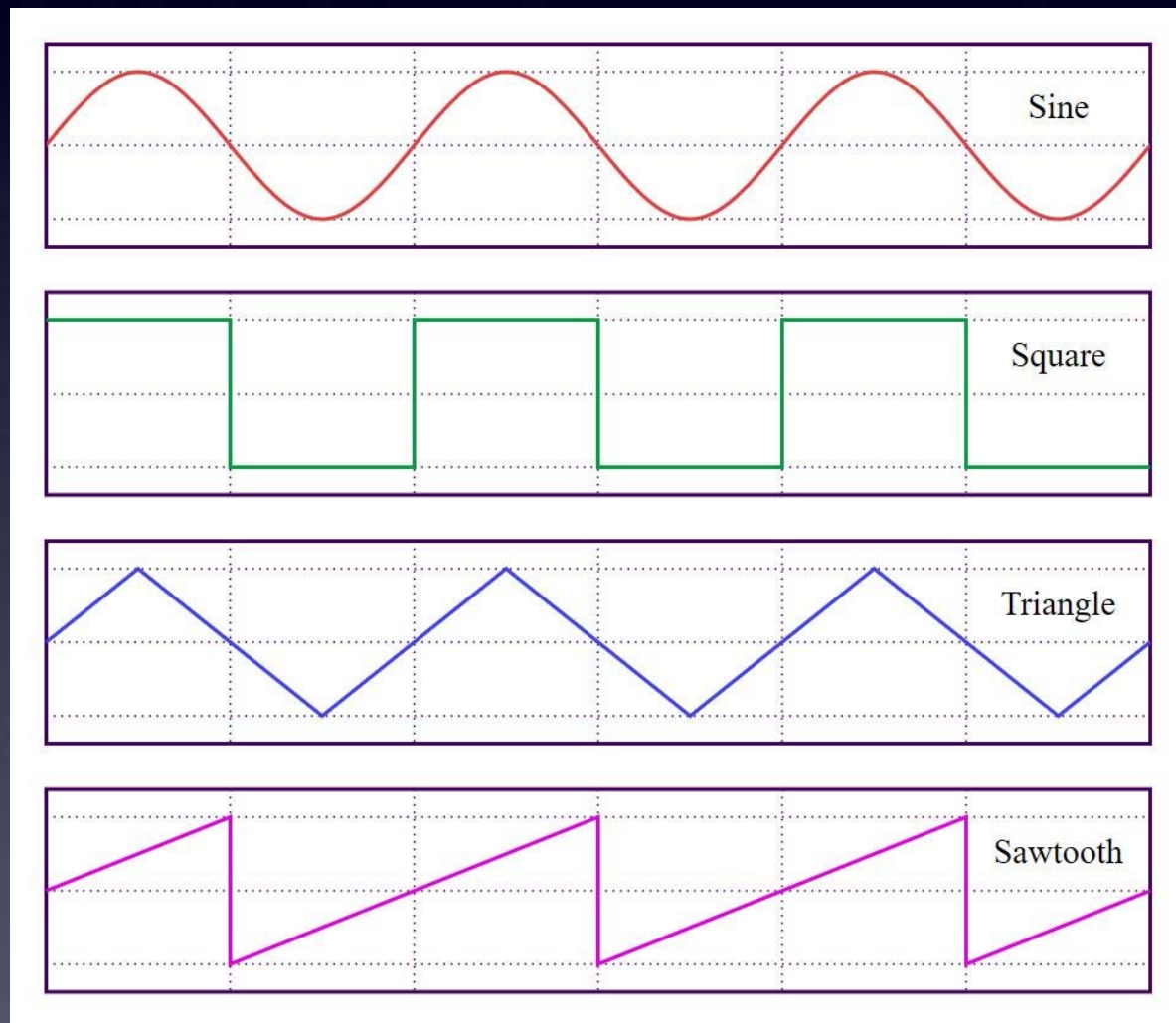
Some Types of Synthesizers

Analog



Some Types of Synthesizers

Analog



Modular Analog Synthesizer:

- Basic waveform oscillators
- Filters (to muck with sound)

Some Types of Synthesizers

Digital



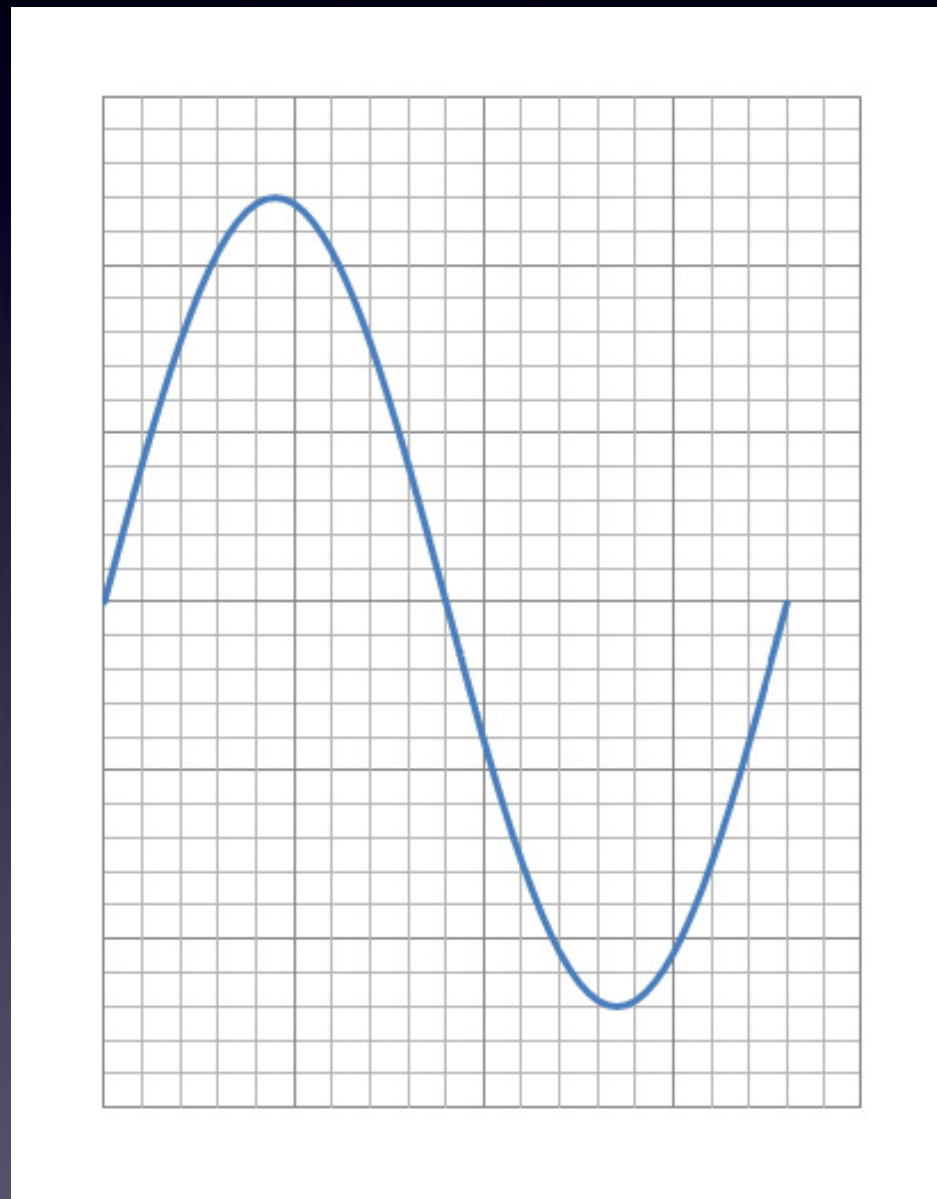
Some Types of Synthesizers

Digital

- Break things into little bits (or create little bits)
- Mess with it
- Put it back together again

Digital Signal Processing

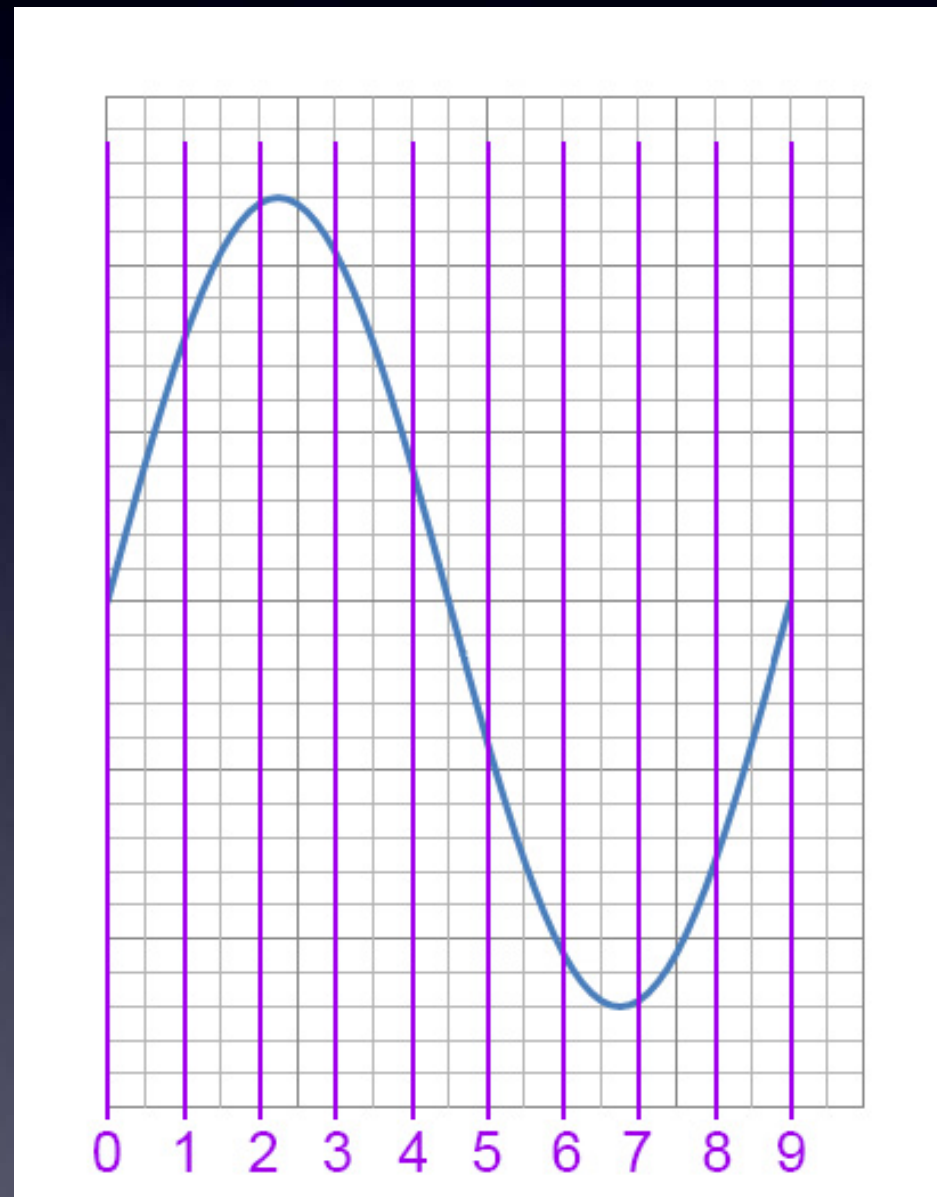
Analog waveform



Digital Signal Processing

To record it digitally

First slice it
(equal time slices)

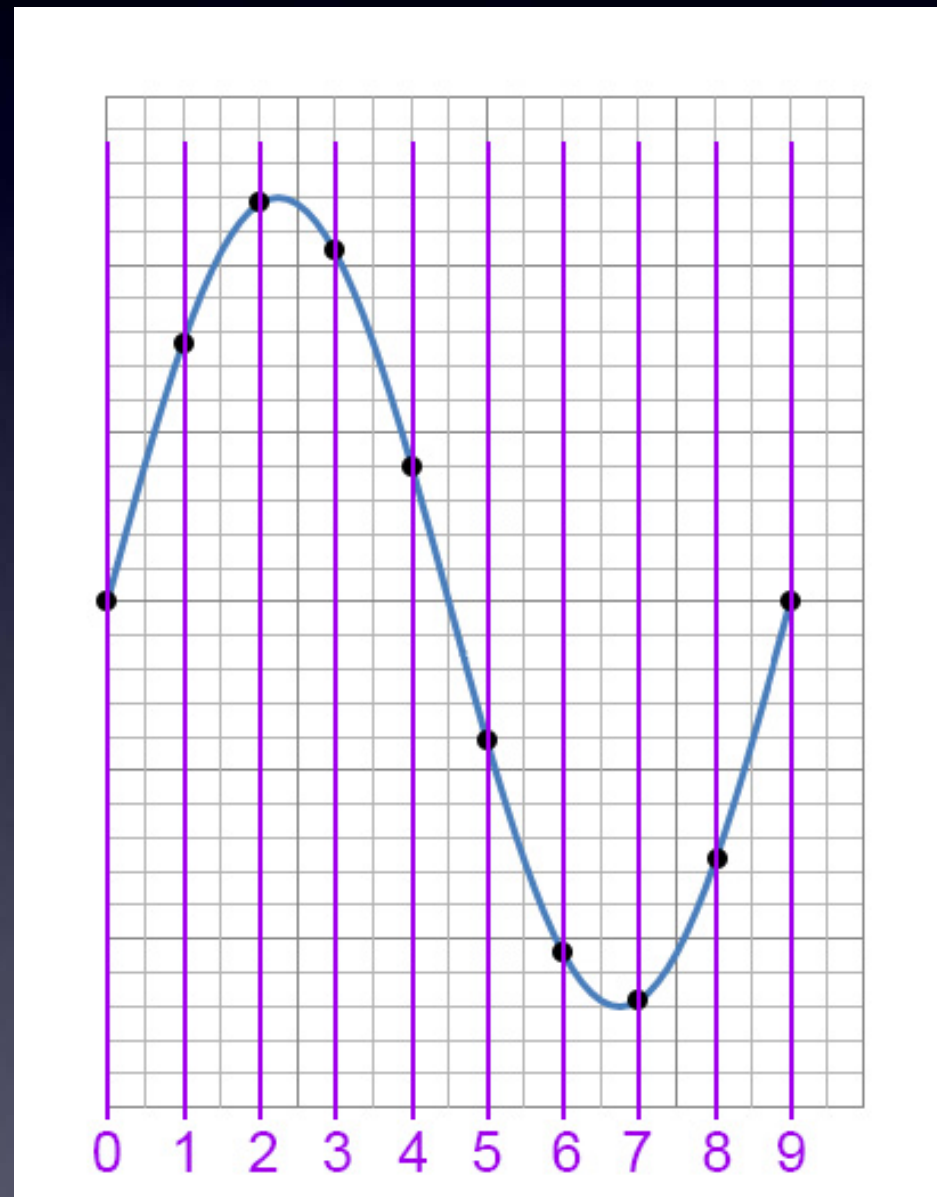


Digital Signal Processing

To record it digitally

First slice it

Then get the values

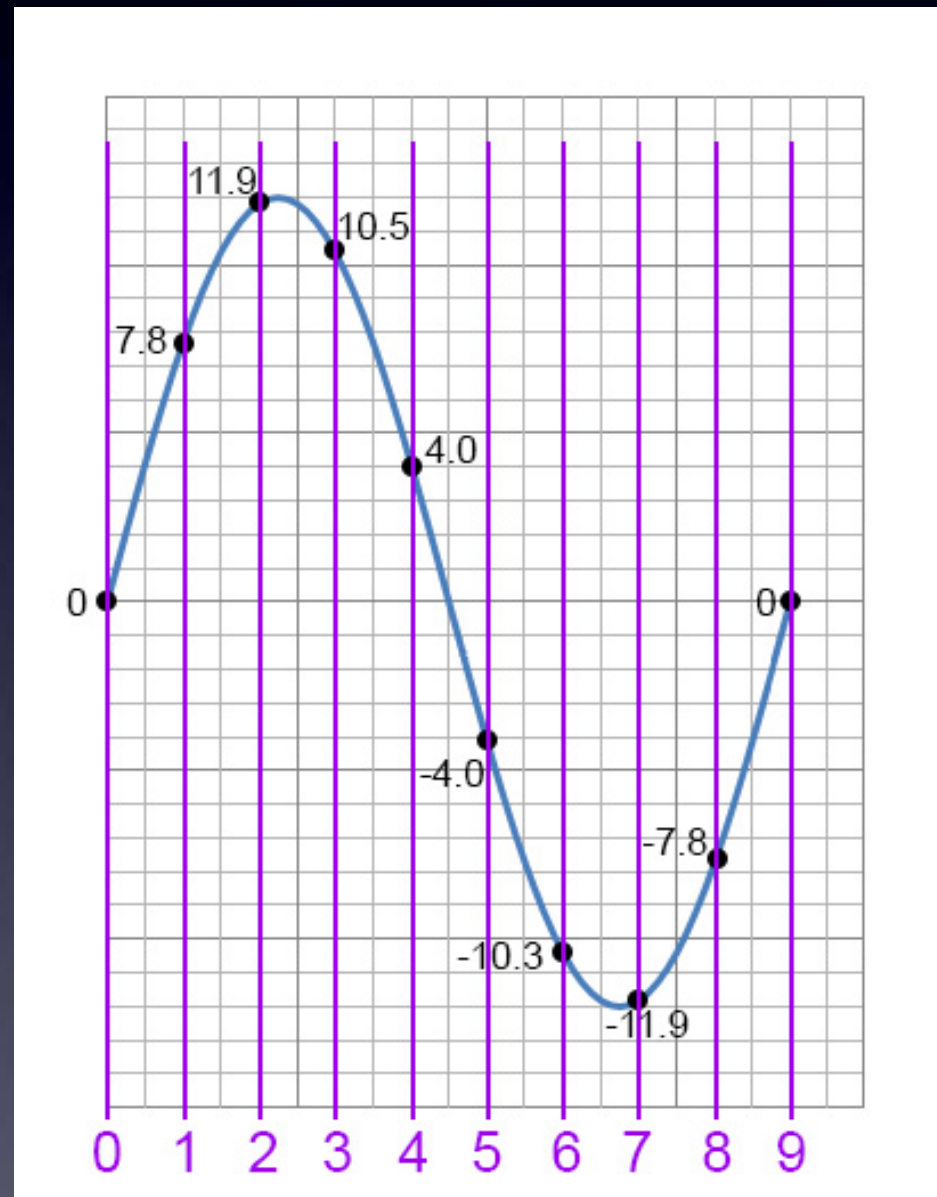


Digital Signal Processing

To record it digitally

First slice it

Then get the values



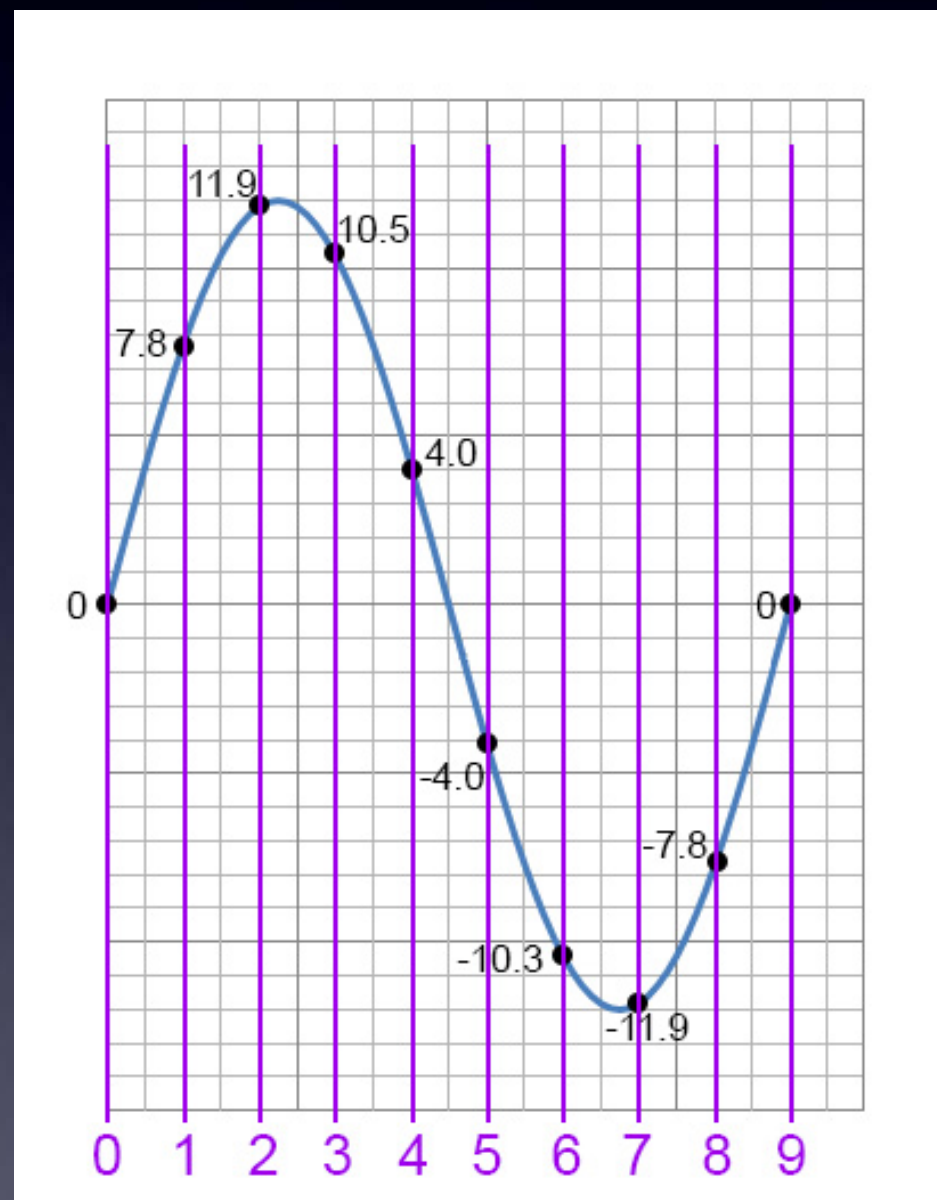
Digital Signal Processing

To record it digitally

First slice it

Then get the values

Then store the values



Sample #	Digitized Value
0	0.0
1	7.8
2	11.9
3	10.5
4	4.0
5	-4.0
6	-10.3
7	-11.9
8	-7.8
9	0.0

Digital Signal Processing

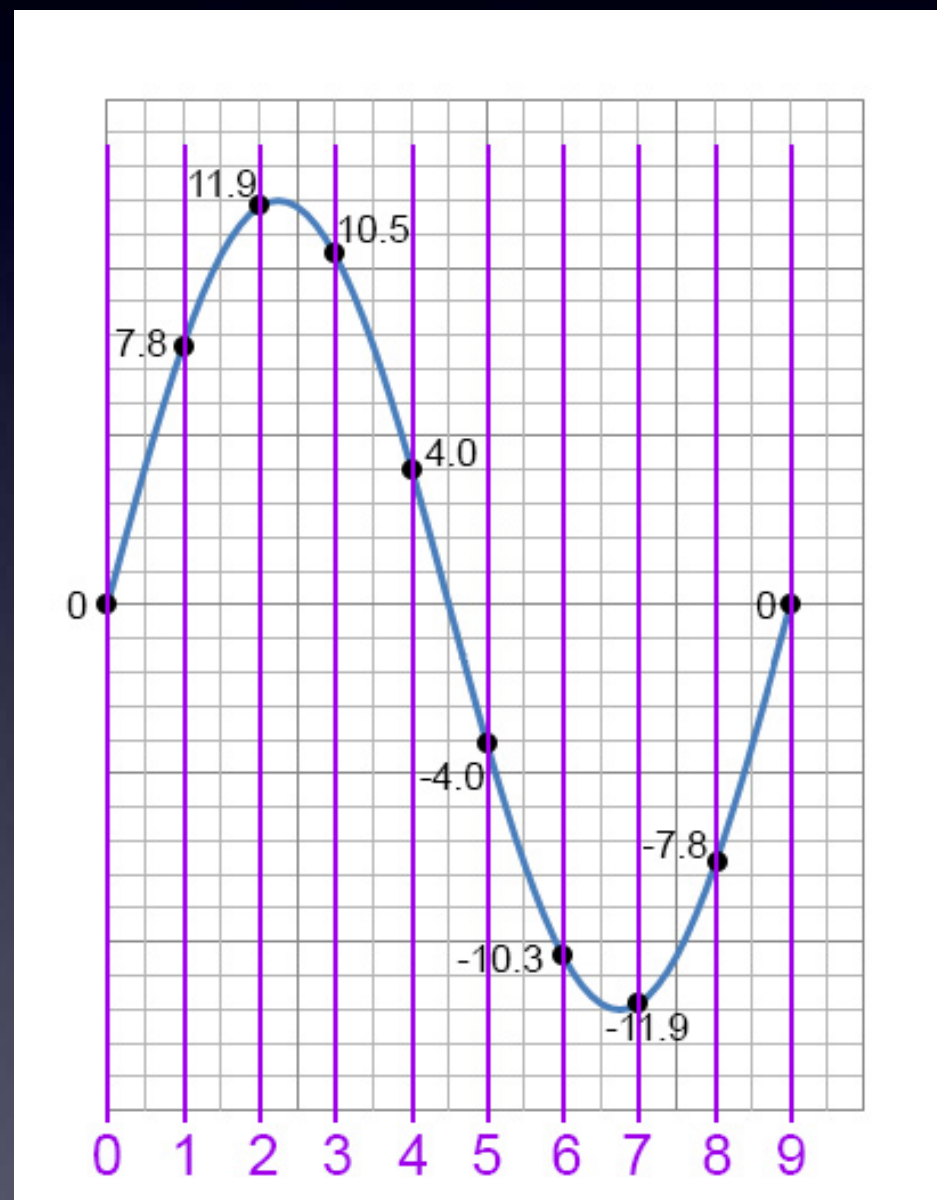
To record it digitally

First slice it

Then get the values

Then store the values

*This is called:
"Sampling"*



Sample #	Digitized Value
0	0.0
1	7.8
2	11.9
3	10.5
4	4.0
5	-4.0
6	-10.3
7	-11.9
8	-7.8
9	0.0

Digital Signal Processing

Samples stored
in
Memory

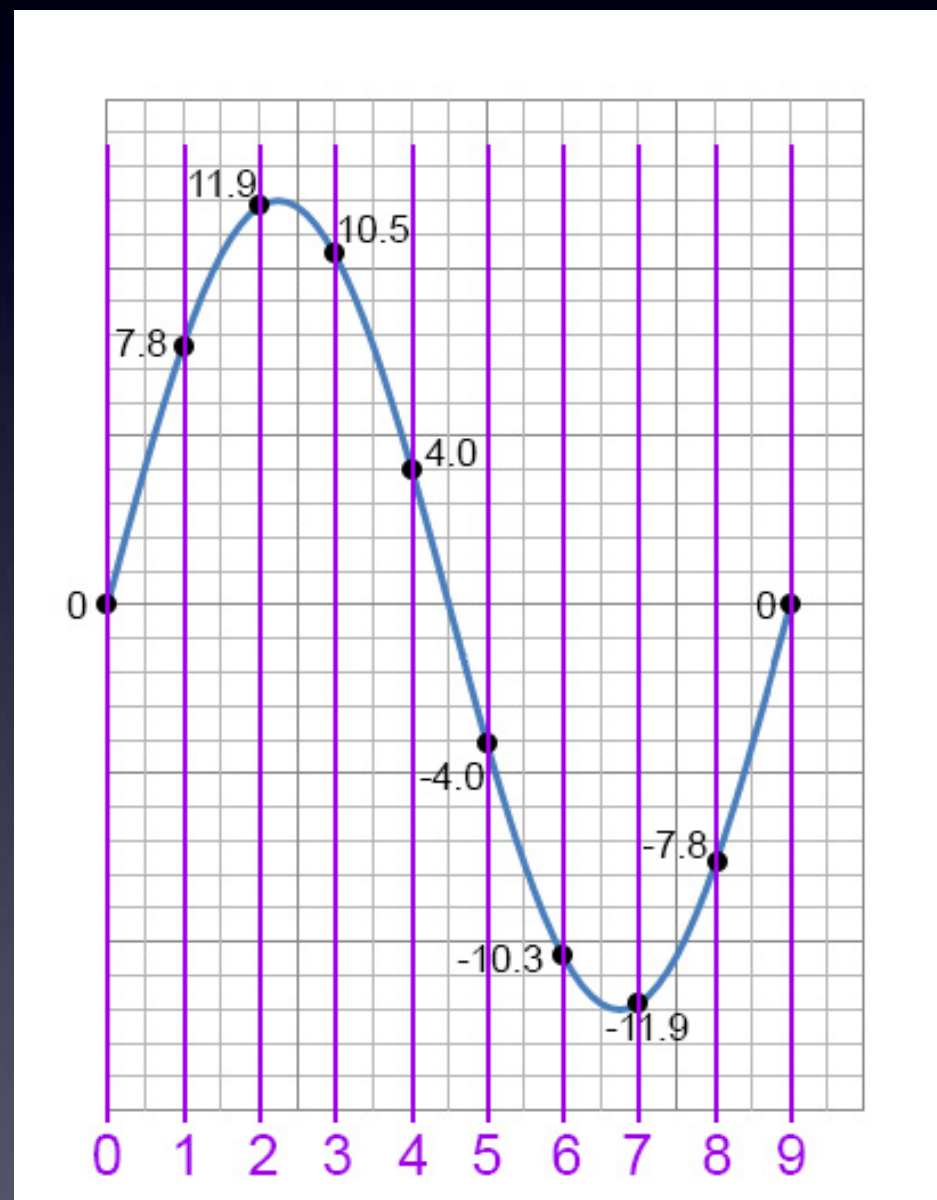
To record it digitally

First slice it

Then get the values

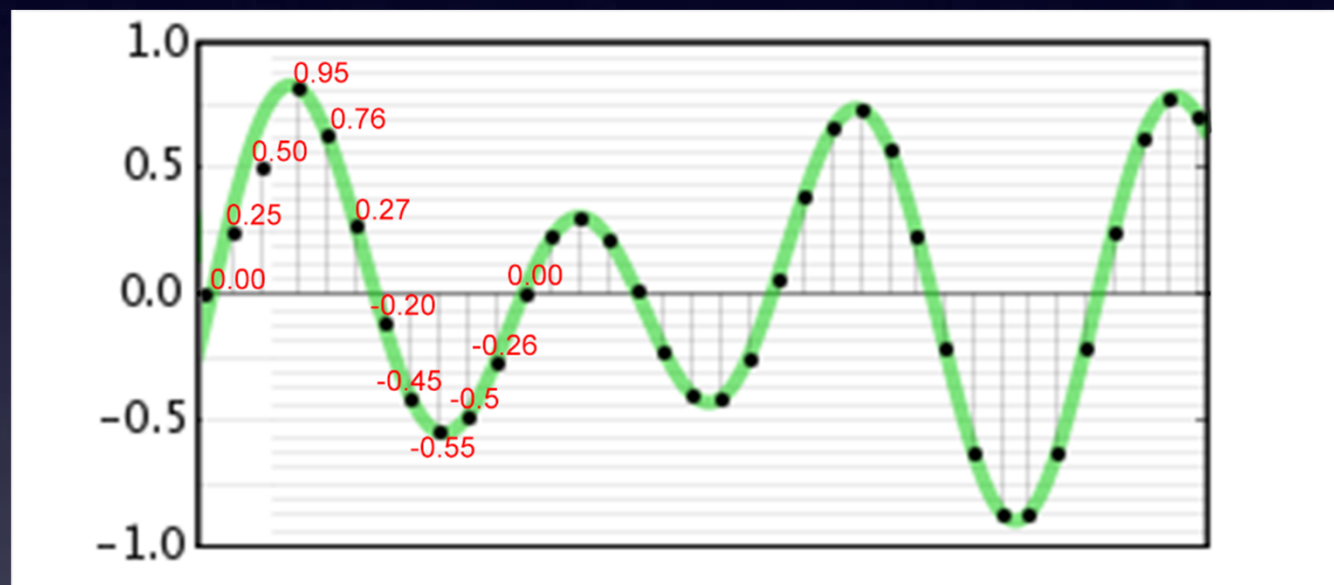
Then store the values

Waveform is:
"Digitized"



Memory loc	Memory contents
0	0.0
1	7.8
2	11.9
3	10.5
4	4.0
5	-4.0
6	-10.3
7	-11.9
8	-7.8
9	0.0

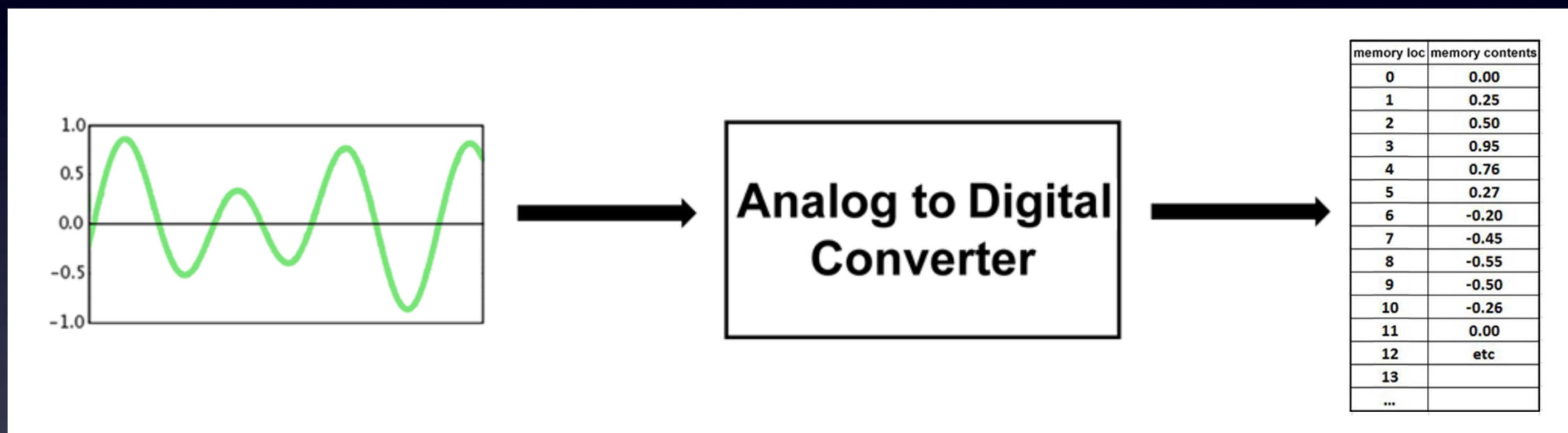
Digital Signal Processing



memory loc	memory contents
0	0.00
1	0.25
2	0.50
3	0.95
4	0.76
5	0.27
6	-0.20
7	-0.45
8	-0.55
9	-0.50
10	-0.26
11	0.00
12	etc
13	
...	

Digitized waveform can be any soundwave

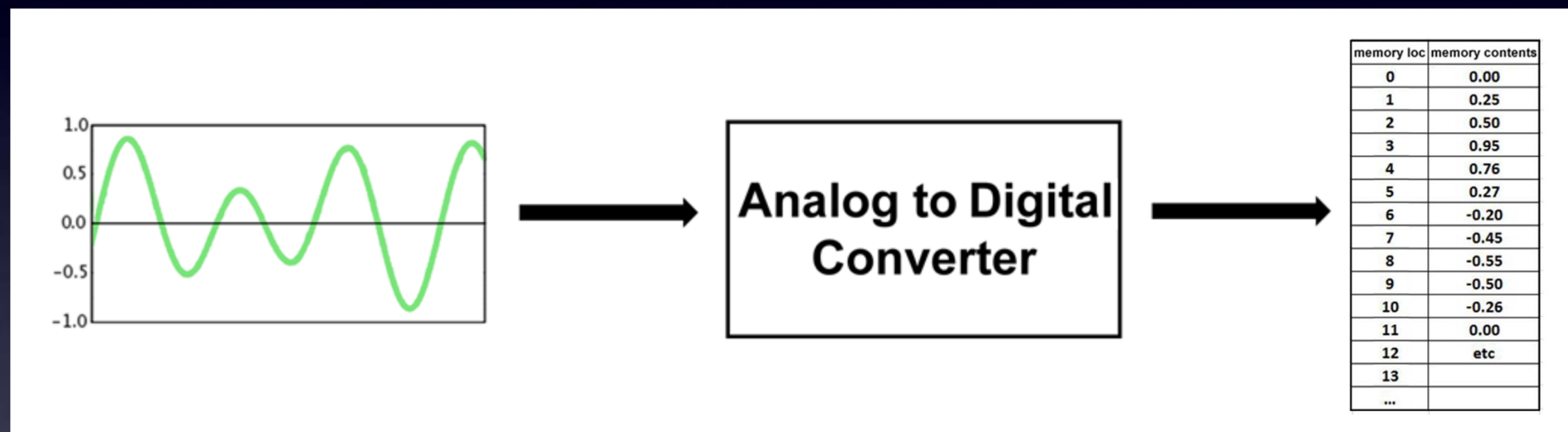
Digital Signal Processing



Analog to Digital Conversion:

sampling an analog waveform to store it in digital memory

Digital Signal Processing



A/D

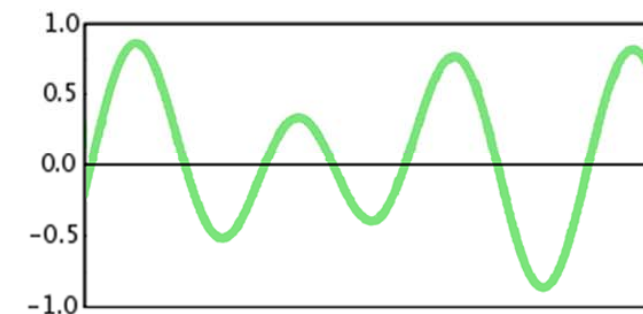
sampling an analog waveform to store it in digital memory

Digital Signal Processing

How do we play back a digitized waveform?

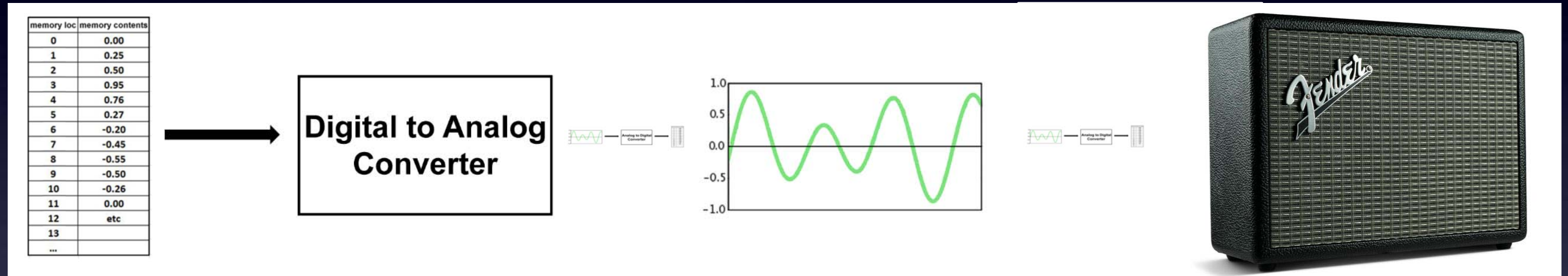
Digital Signal Processing

memory loc	memory contents
0	0.00
1	0.25
2	0.50
3	0.95
4	0.76
5	0.27
6	-0.20
7	-0.45
8	-0.55
9	-0.50
10	-0.26
11	0.00
12	etc
13	
...	



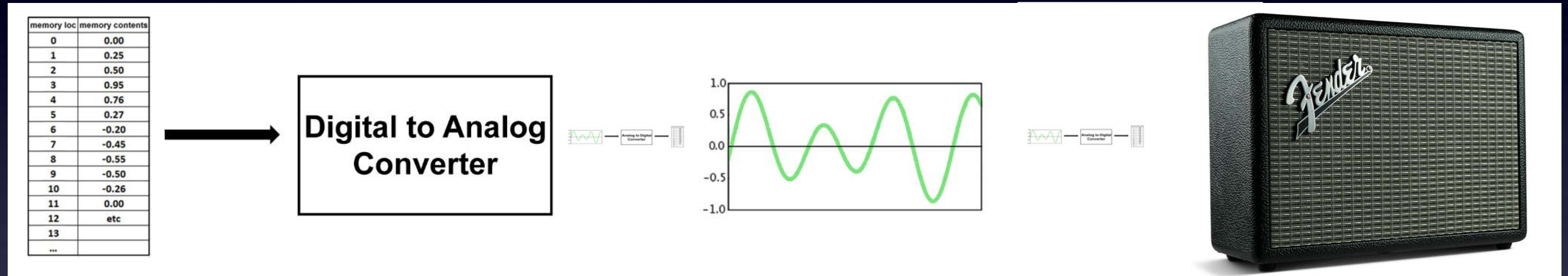
Digital to Analog Conversion:
Playing back the Digitized waveform

Digital Signal Processing



Digital to Analog Conversion:
Playing back the Digitized waveform

Digital Signal Processing



D/A

Playing back the Digitized waveform

Digital Signal Processing

How do you do
D/A ?

Digital Signal Processing

D/A chip (expensive)

or

PWM

Digital Signal Processing

PWM ?



Square Wave:

ON half the time / OFF half of the time

Digital Signal Processing

PWM ?



Square Wave:

ON half the time / OFF half of the time

(half the energy of ON all the time)

Digital Signal Processing

PWM ?



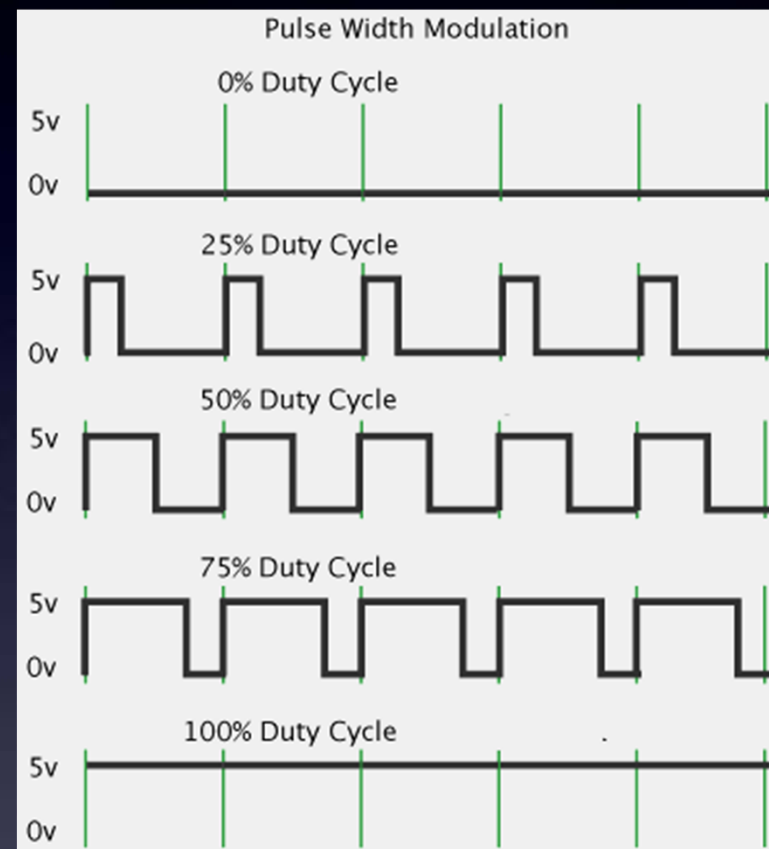
Pulse Wave:

ON and OFF at any ratio you like

This waveform: ON for 25% of the time / OFF for 75% of the time

PWM ?

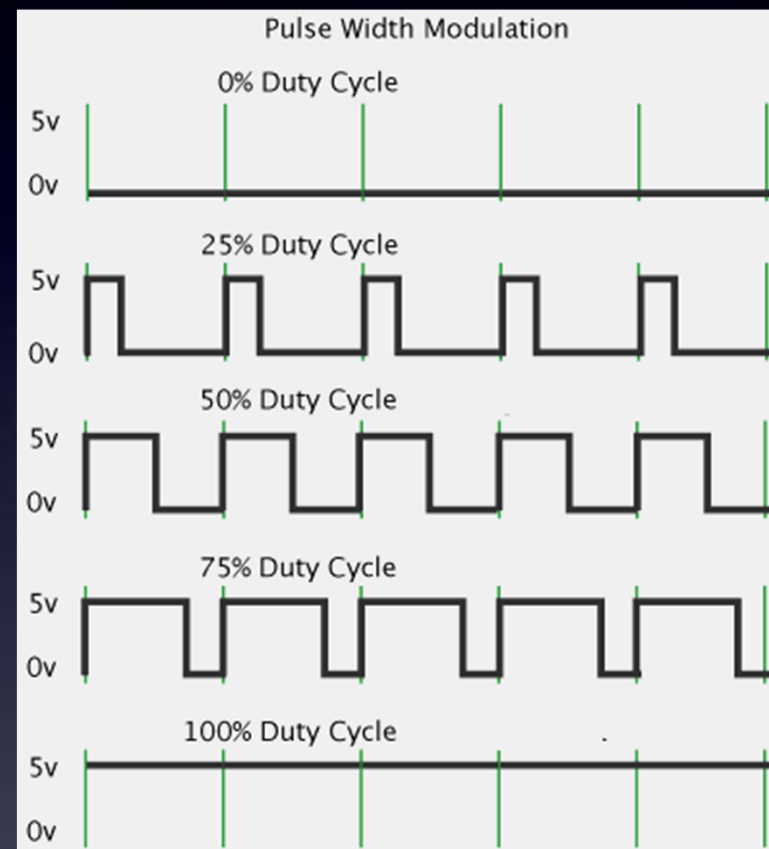
Digital Signal Processing



Pulse Wave:

ON and OFF at any ratio you like

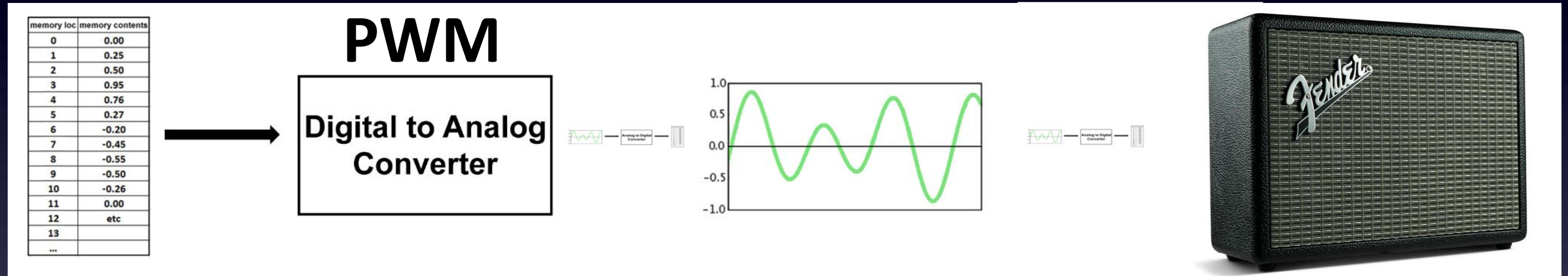
Digital Signal Processing



PWM

Pulse Width Modulation

Digital Signal Processing



D/A

Using PWM for playing back the Digitized waveform

Digital Signal Processing

Kind of complicated to code

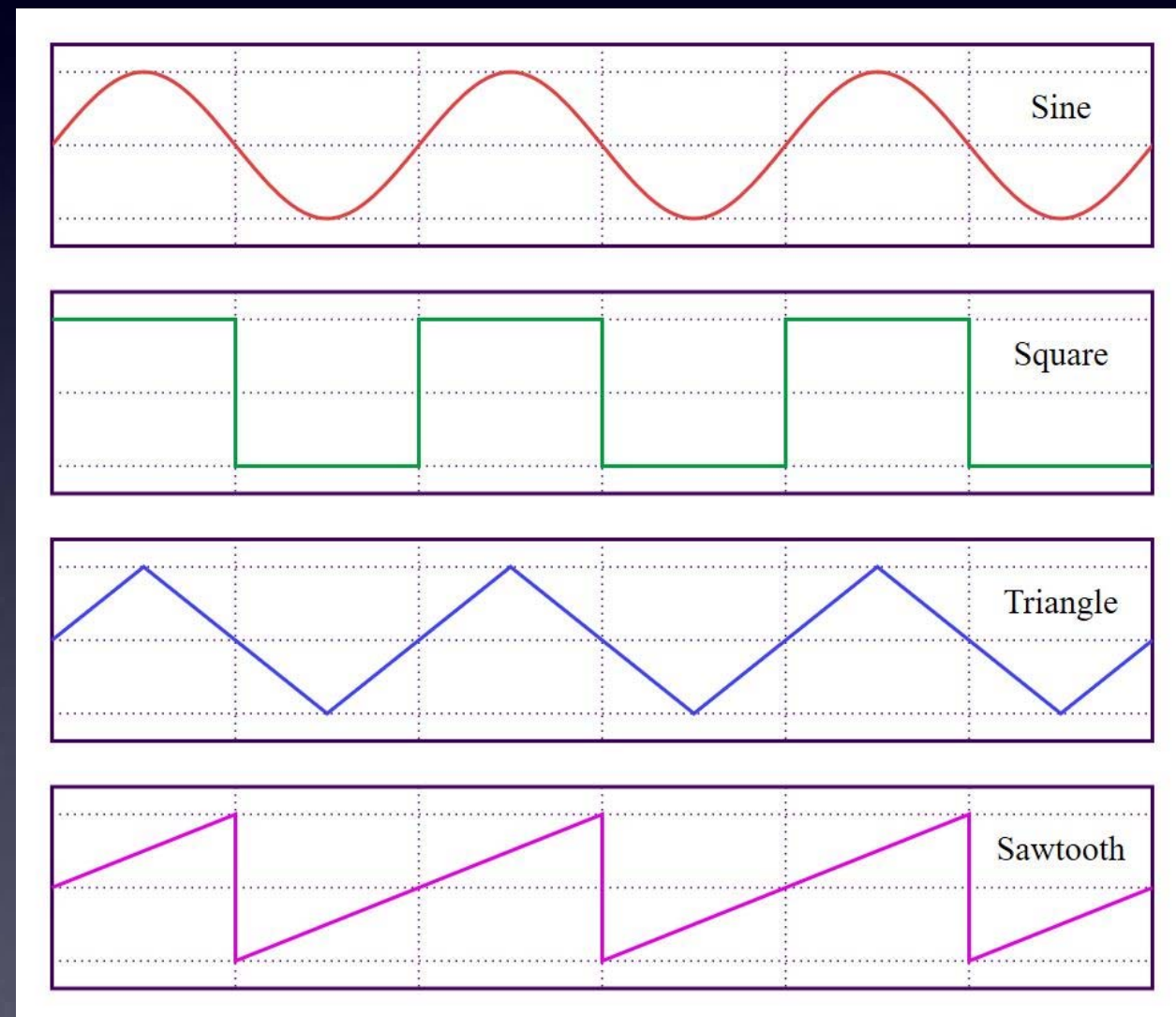
So, my ArduTouch software makes it easy

- Create “oscillators” with a couple lines of code
- Create “dynamics” with a couple lines of code

“Dynamics”
make the sound interesting

Digital Signal Processing

Some “Oscillators”:



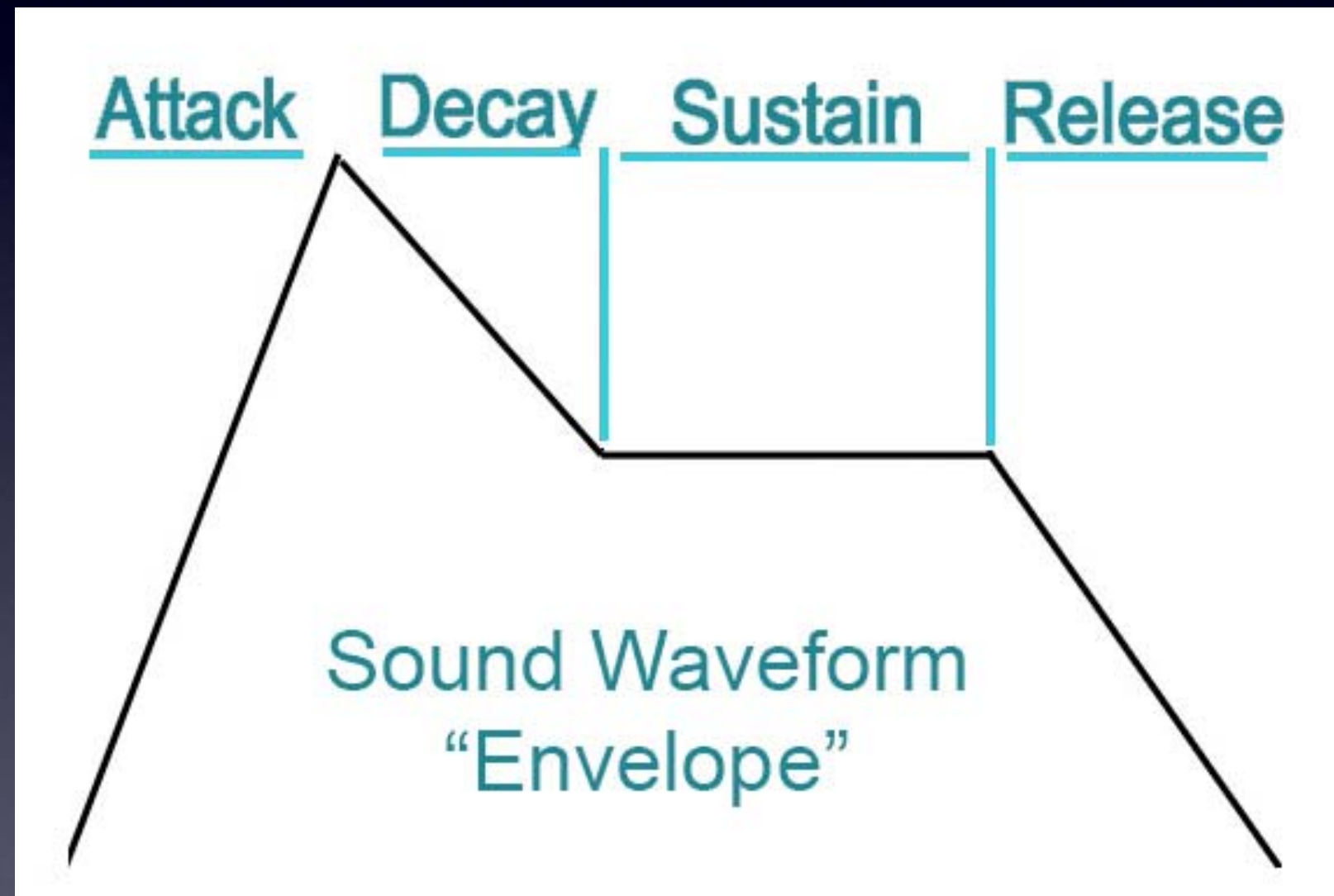
Digital Signal Processing

Some “Dynamics”:

- ADSR
- Tremolo
- Portamento
- Envelopes
- Filters
- Effects

Digital Signal Processing

ADSR:



Digital Signal Processing

Some “Dynamics”:

- ADSR
- Tremolo – *constant changing volume*
- Portamento
- Envelopes
- Filters
- Effects

Digital Signal Processing

Some “Dynamics”:

- ADSR
- Tremolo – *constant changing volume*
- Portamento – *glide between notes*
- Envelopes
- Filters
- Effects

Digital Signal Processing

Some “Dynamics”:

- ADSR
- Tremolo – *constant changing volume*
- Portamento – *glide between notes*
- Envelopes – *beyond ADSR*
- Filters
- Effects

Digital Signal Processing

Some “Dynamics”:

- ADSR
- Tremolo – *constant changing volume*
- Portamento – *glide between notes*
- Envelopes – *beyond ADSR*
- Filters – *like bass & treble – subtle to crazy*
- Effects

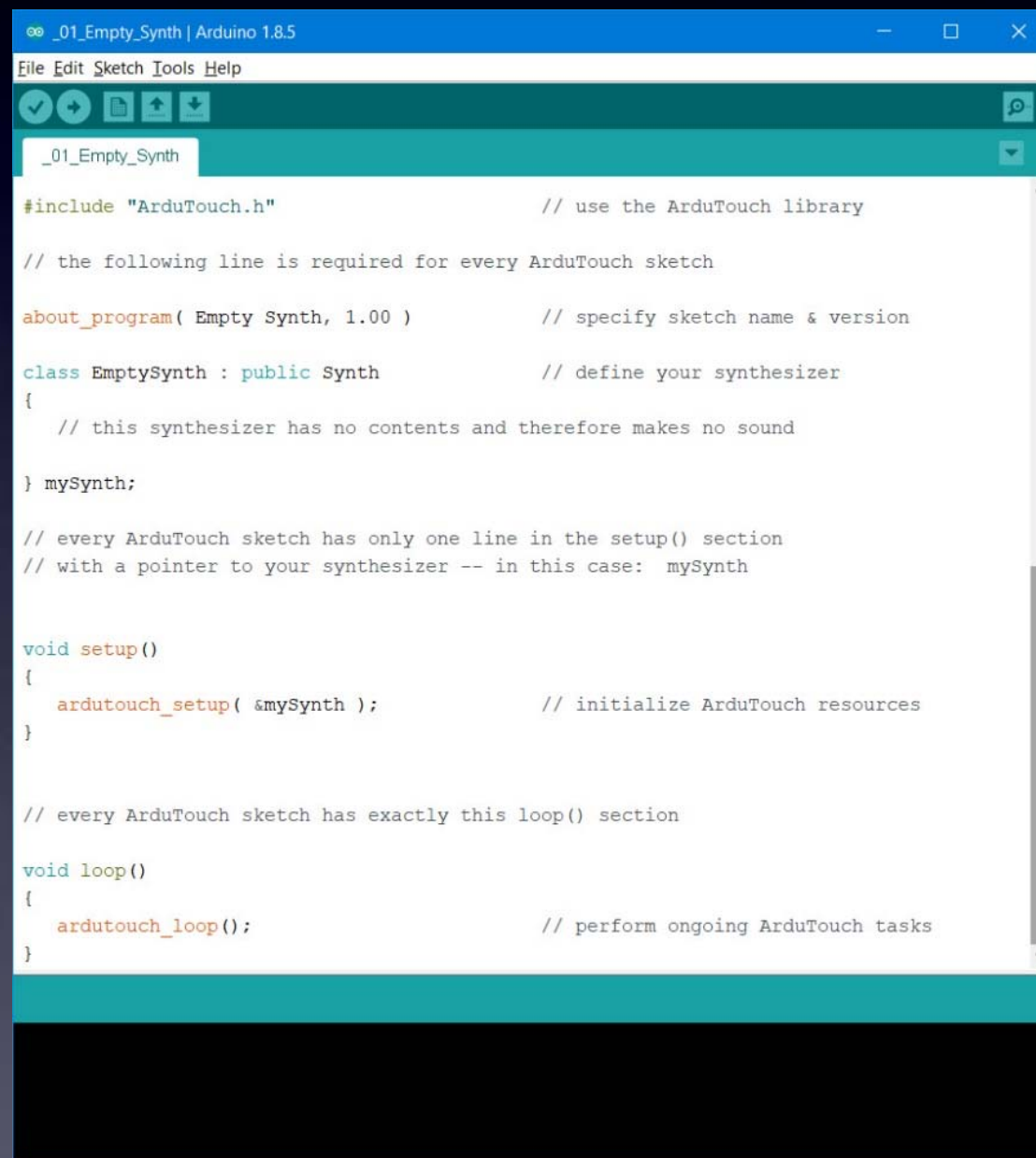
Digital Signal Processing

Some “Dynamics”:

- ADSR
- Tremolo – *constant changing volume*
- Portamento – *glide between notes*
- Envelopes – *beyond ADSR*
- Filters – *like bass & treble – subtle to crazy*
- Effects – *mess with the sound!*

ArduTouch

Arduino-Compatible

A screenshot of the Arduino IDE interface. The title bar reads "_01_Empty_Synth | Arduino 1.8.5". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for opening, saving, and running. A tab labeled "_01_Empty_Synth" is active. The main text area contains the following C++ code:

```
#include "ArduTouch.h"           // use the ArduTouch library

// the following line is required for every ArduTouch sketch

about_program( Empty Synth, 1.00 )    // specify sketch name & version

class EmptySynth : public Synth       // define your synthesizer
{
    // this synthesizer has no contents and therefore makes no sound
} mySynth;

// every ArduTouch sketch has only one line in the setup() section
// with a pointer to your synthesizer -- in this case: mySynth

void setup()
{
    ardutouch_setup( &mySynth );      // initialize ArduTouch resources
}

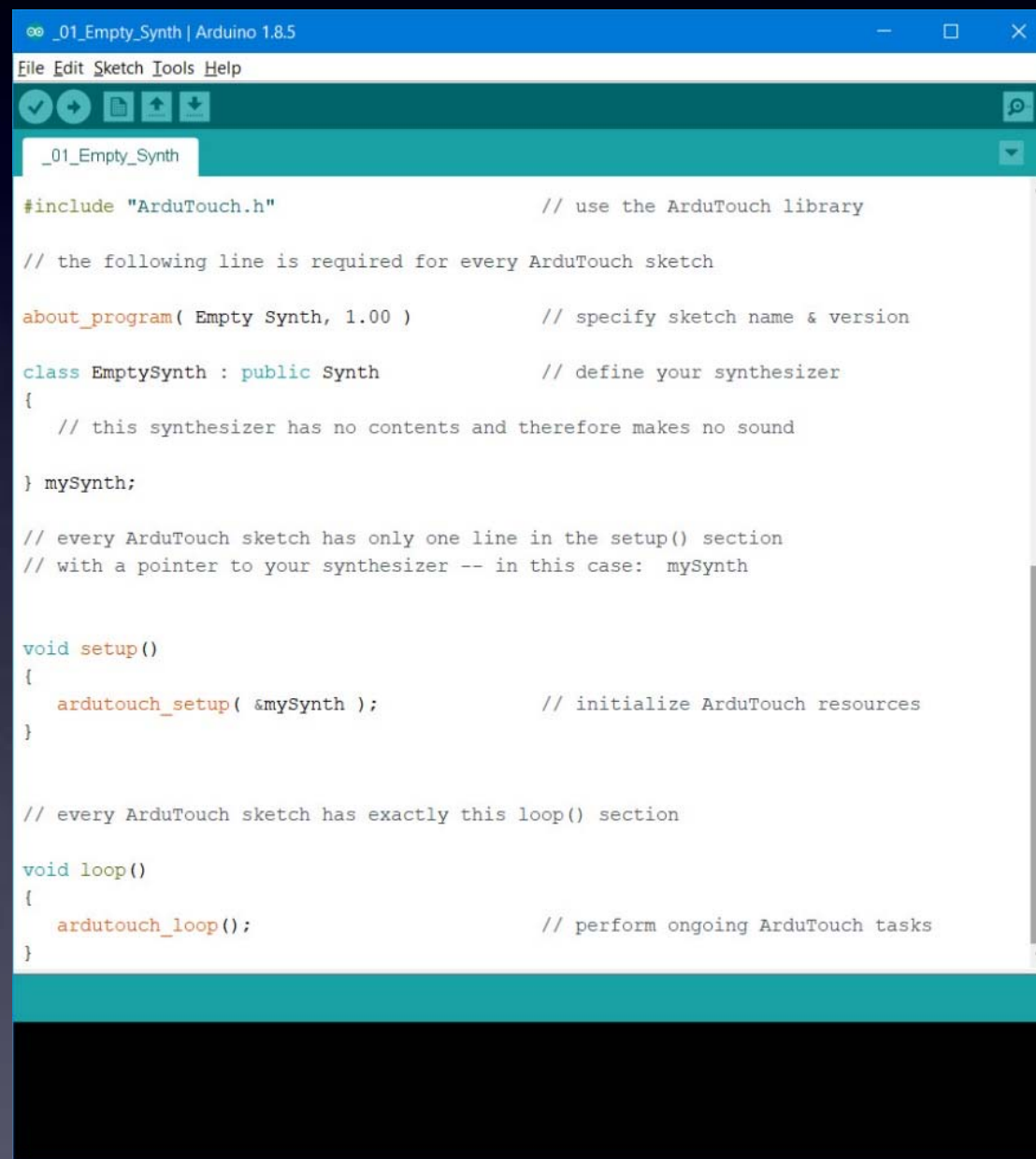
// every ArduTouch sketch has exactly this loop() section

void loop()
{
    ardutouch_loop();                 // perform ongoing ArduTouch tasks
}
```

With
Tutorial examples

ArduTouch

Arduino-Compatible



```

_01_Empty_Synth | Arduino 1.8.5
File Edit Sketch Tools Help

_01_Empty_Synth

#include "ArduTouch.h"           // use the ArduTouch library

// the following line is required for every ArduTouch sketch

about_program( Empty Synth, 1.00 )    // specify sketch name & version

class EmptySynth : public Synth       // define your synthesizer
{
    // this synthesizer has no contents and therefore makes no sound

} mySynth;

// every ArduTouch sketch has only one line in the setup() section
// with a pointer to your synthesizer -- in this case: mySynth

void setup()
{
    ardutouch_setup( &mySynth );      // initialize ArduTouch resources
}

// every ArduTouch sketch has exactly this loop() section

void loop()
{
    ardutouch_loop();                 // perform ongoing ArduTouch tasks
}

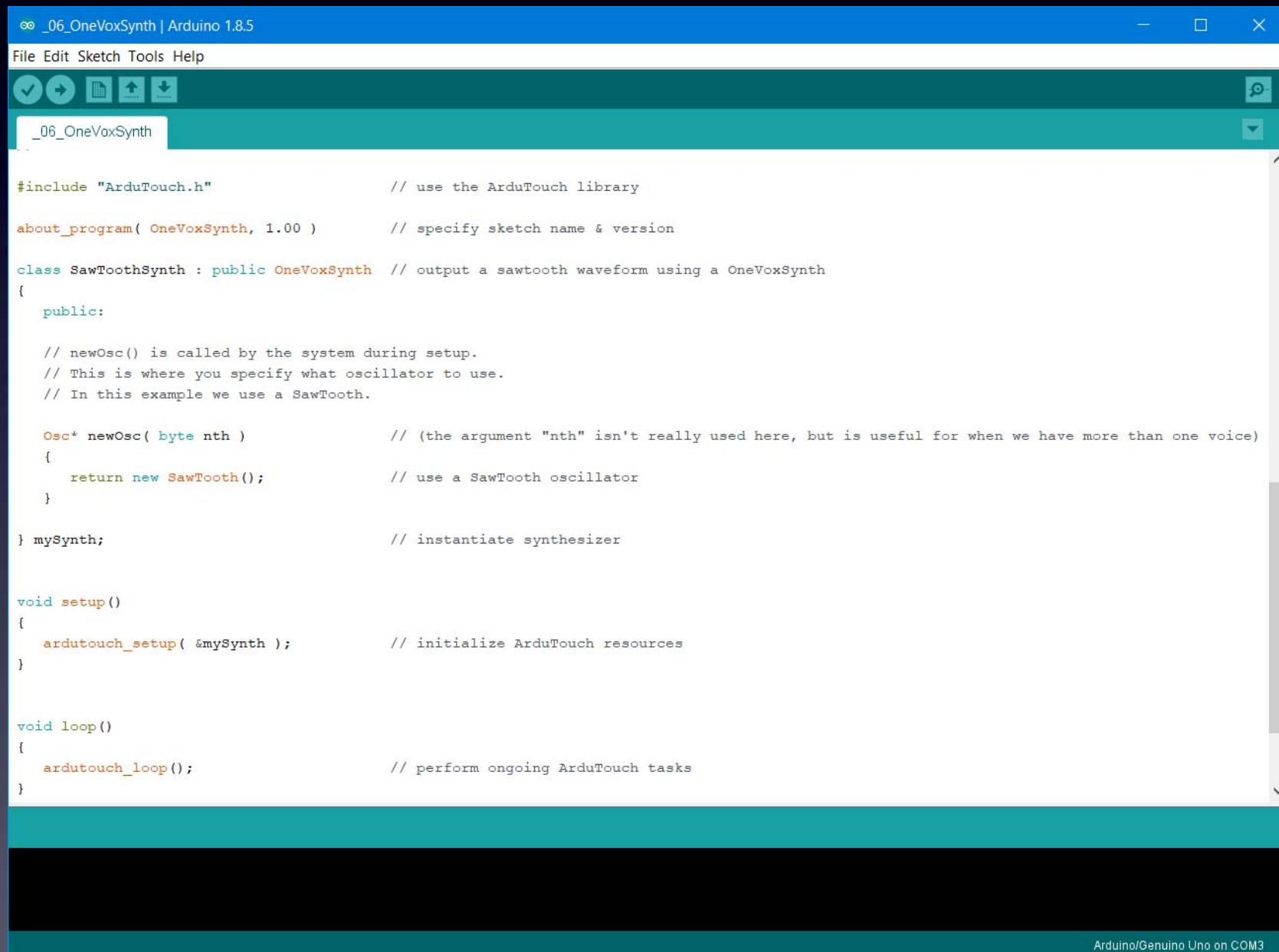
```

**With
Tutorial examples**

*Follow examples
01 through 09
to easily learn
to code your own
synthesizers*

ArduTouch

Arduino-Compatible



```
_06_OneVoxSynth | Arduino 1.8.5
File Edit Sketch Tools Help

#include "ArduTouch.h"           // use the ArduTouch library

about_program( OneVoxSynth, 1.00 ) // specify sketch name & version

class SawToothSynth : public OneVoxSynth // output a sawtooth waveform using a OneVoxSynth
{
public:

// newOsc() is called by the system during setup.
// This is where you specify what oscillator to use.
// In this example we use a SawTooth.

Osc* newOsc( byte nth )           // (the argument "nth" isn't really used here, but is useful for when we have more than one voice)
{
return new SawTooth();           // use a SawTooth oscillator
}

} mySynth;                        // instantiate synthesizer

void setup()
{
ardutouch_setup( &mySynth );     // initialize ArduTouch resources
}

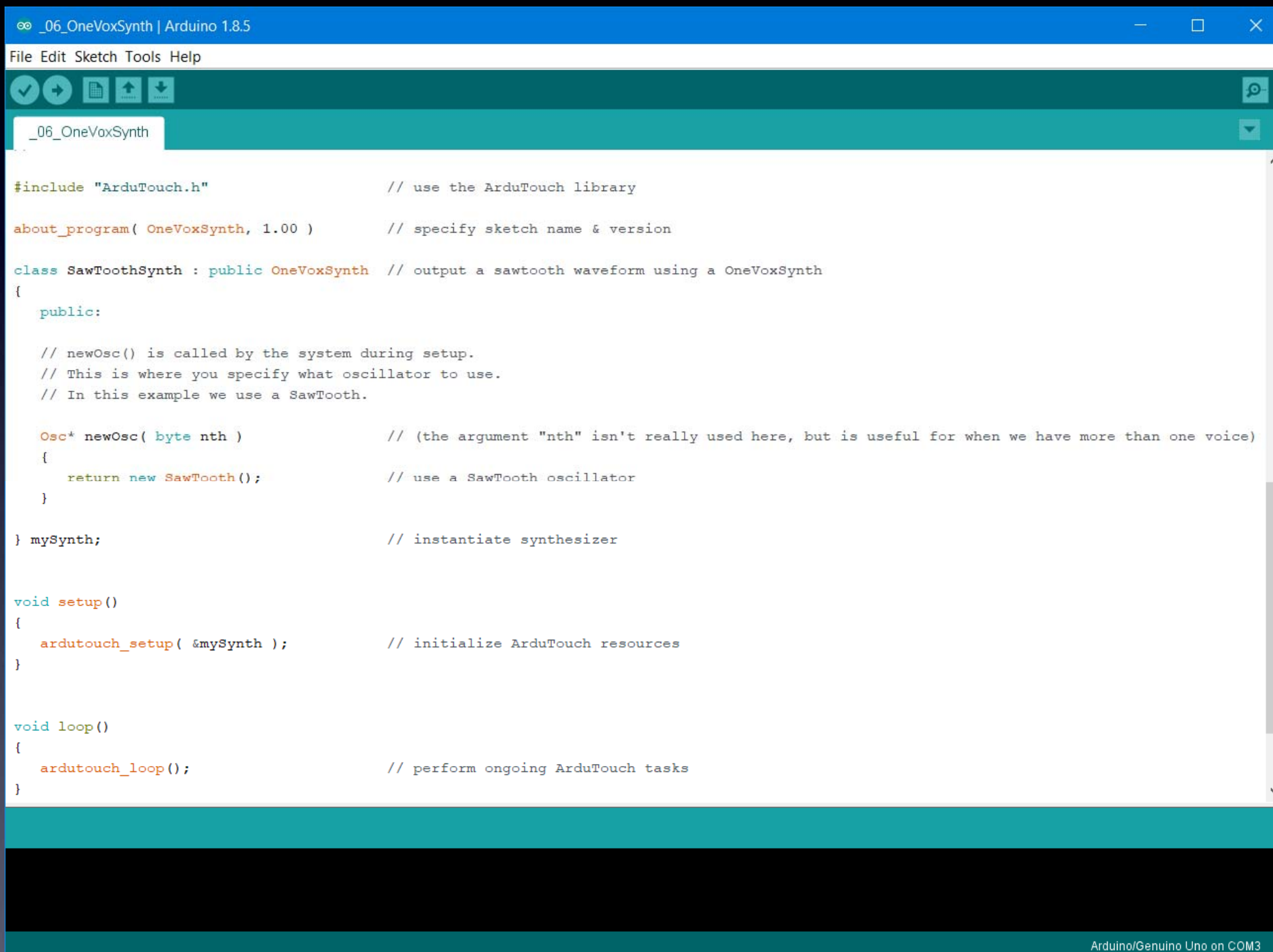
void loop()
{
ardutouch_loop();                // perform ongoing ArduTouch tasks
}

Arduino/Genuino Uno on COM3
```

**With
extensive
Arduino library
for ArduTouch**

*to make it easy
to create
your own synths*

ArduTouch



```
#include "ArduTouch.h"           // use the ArduTouch library

about_program( OneVoxSynth, 1.00 ) // specify sketch name & version

class SawToothSynth : public OneVoxSynth // output a sawtooth waveform using a OneVoxSynth
{
public:

    // newOsc() is called by the system during setup.
    // This is where you specify what oscillator to use.
    // In this example we use a SawTooth.

    Osc* newOsc( byte nth ) // (the argument "nth" isn't really used here, but is useful for when we have more than one voice)
    {
        return new SawTooth(); // use a SawTooth oscillator
    }

} mySynth; // instantiate synthesizer

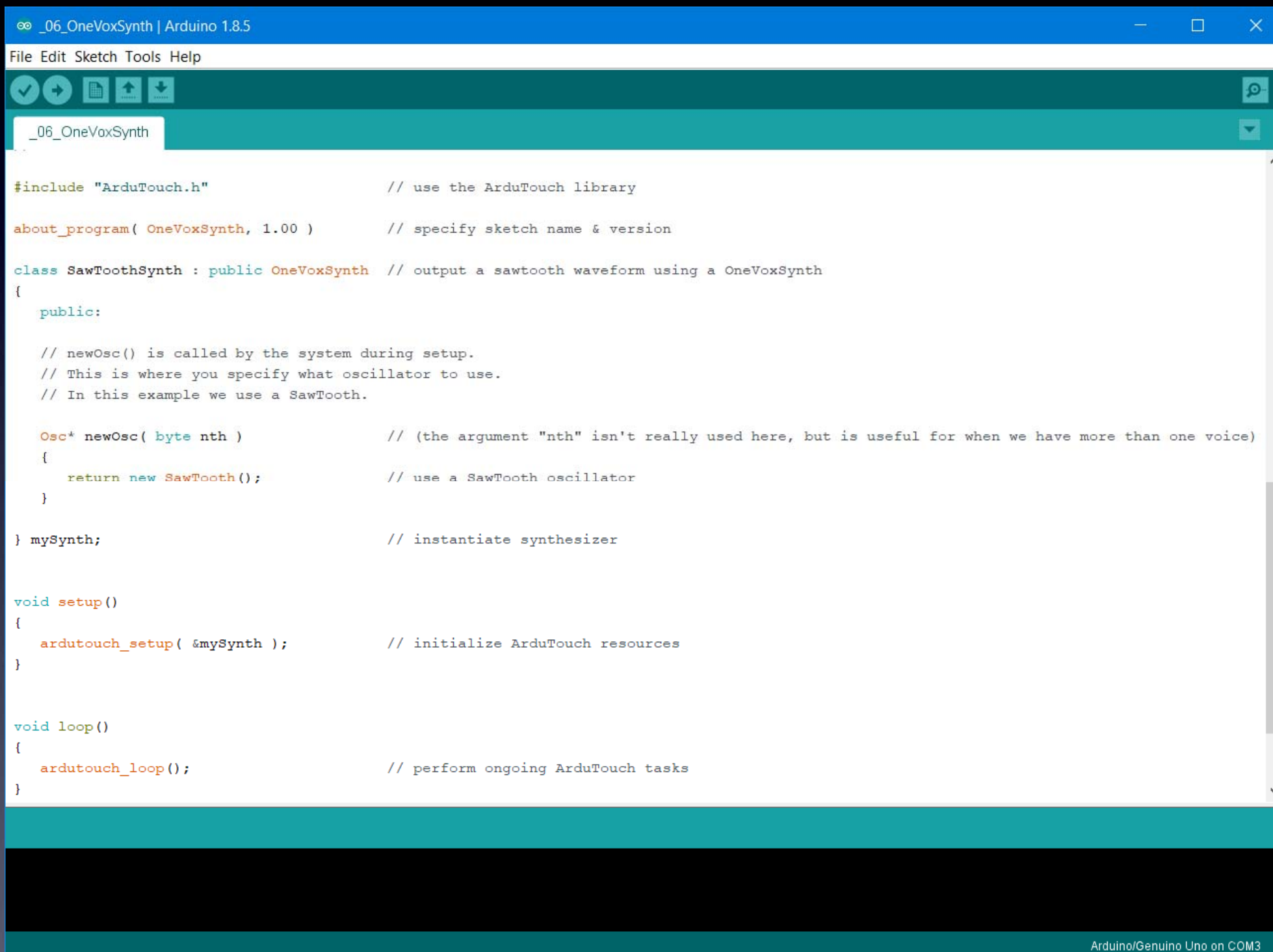
void setup()
{
    ardutouch_setup( &mySynth ); // initialize ArduTouch resources
}

void loop()
{
    ardutouch_loop(); // perform ongoing ArduTouch tasks
}
```

Complete code for:

- sawtooth waves
- play with keyboard
- change octaves
- volume control

ArduTouch



```
Arduino IDE: _06_OneVoxSynth | Arduino 1.8.5
File Edit Sketch Tools Help

#include "ArduTouch.h"           // use the ArduTouch library

about_program( OneVoxSynth, 1.00 ) // specify sketch name & version

class SawToothSynth : public OneVoxSynth // output a sawtooth waveform using a OneVoxSynth
{
public:

// newOsc() is called by the system during setup.
// This is where you specify what oscillator to use.
// In this example we use a SawTooth.

Osc* newOsc( byte nth )           // (the argument "nth" isn't really used here, but is useful for when we have more than one voice)
{
    return new SawTooth();         // use a SawTooth oscillator
}

} mySynth;                        // instantiate synthesizer

void setup()
{
    ardutouch_setup( &mySynth );   // initialize ArduTouch resources
}

void loop()
{
    ardutouch_loop();              // perform ongoing ArduTouch tasks
}
```

Arduino/Genuino Uno on COM3

Easy to add:

- Tremolo
- Portamento
- Envelopes
- Filters
- Effects
- Other waveforms

ArduTouch



File Edit View History Bookmarks Tools Help

maltman23/ArduTouch: A... X +

GitHub, Inc. (US) | https://github.com/maltman23/ArduTouch

Search

This repository Search Pull requests Issues Gist

maltman23 / ArduTouch

Unwatch 4 Star 9 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

ArduTouch is an Arduino-compatible music synthesizer kit. Build it, and it works! Way low cost (target price per kit is \$25.) It comes with a pre-programmed music synthesizer that makes way cool sounds and music and noise. An ArduTouch library is available for programming in more super nice synthesizer features. For those who want to learn more,... — Edit

7 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

maltman23 update to PCB v1.2 rev C & DuoPoly v2.05 Latest commit b5109e5 on Jul 10		
Arduino	update to PCB v1.2 rev C & DuoPoly v2.05	5 months ago
AssemblyInstructions	ArduTouch assembly instructions	7 months ago
BOM	ArduTouch BOM	7 months ago
Eagle	update to PCB v1.2 rev C & DuoPoly v2.05	5 months ago
Schematic	update to PCB v1.2 rev C & DuoPoly v2.05	5 months ago
.gitattributes	Added .gitattributes & .gitignore files	7 months ago
.gitignore	Added .gitattributes & .gitignore files	7 months ago
README.md	Create README.md	7 months ago

https://github.com/maltman23/ArduTouch/commit/9f3176d57b9e8ad6b8732f00a104f6c858662ef7

Open Hardware – everything is on Github
maltman23

(Don't bring these home)

Tools



Learn To Solder



The following photos will show you how to solder.

But feel free to download the “Soldering Is Easy” comic book for free!

(In many different languages.)

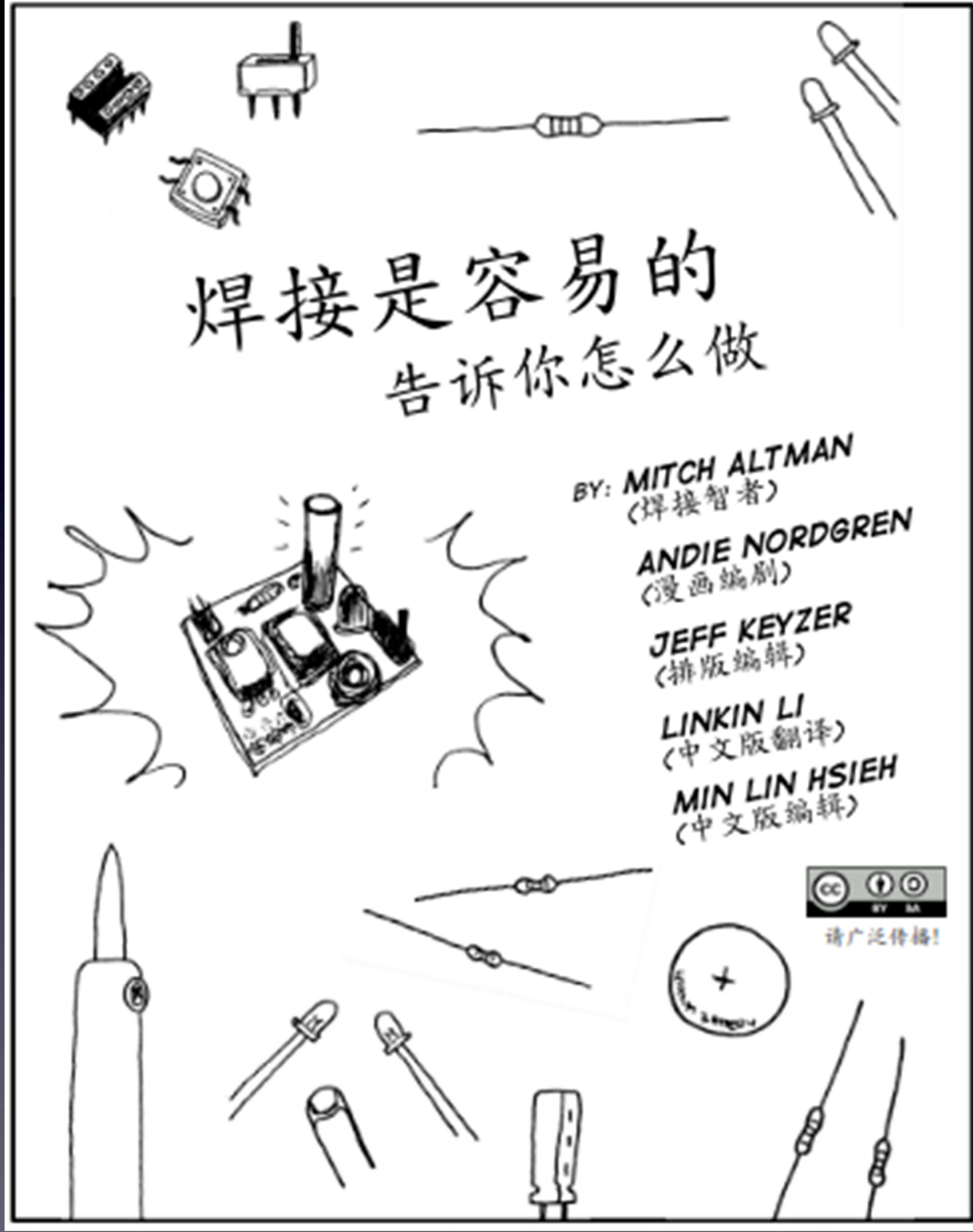
download for free at:
<http://mightyohm.com/soldercomic>

Learn To Solder



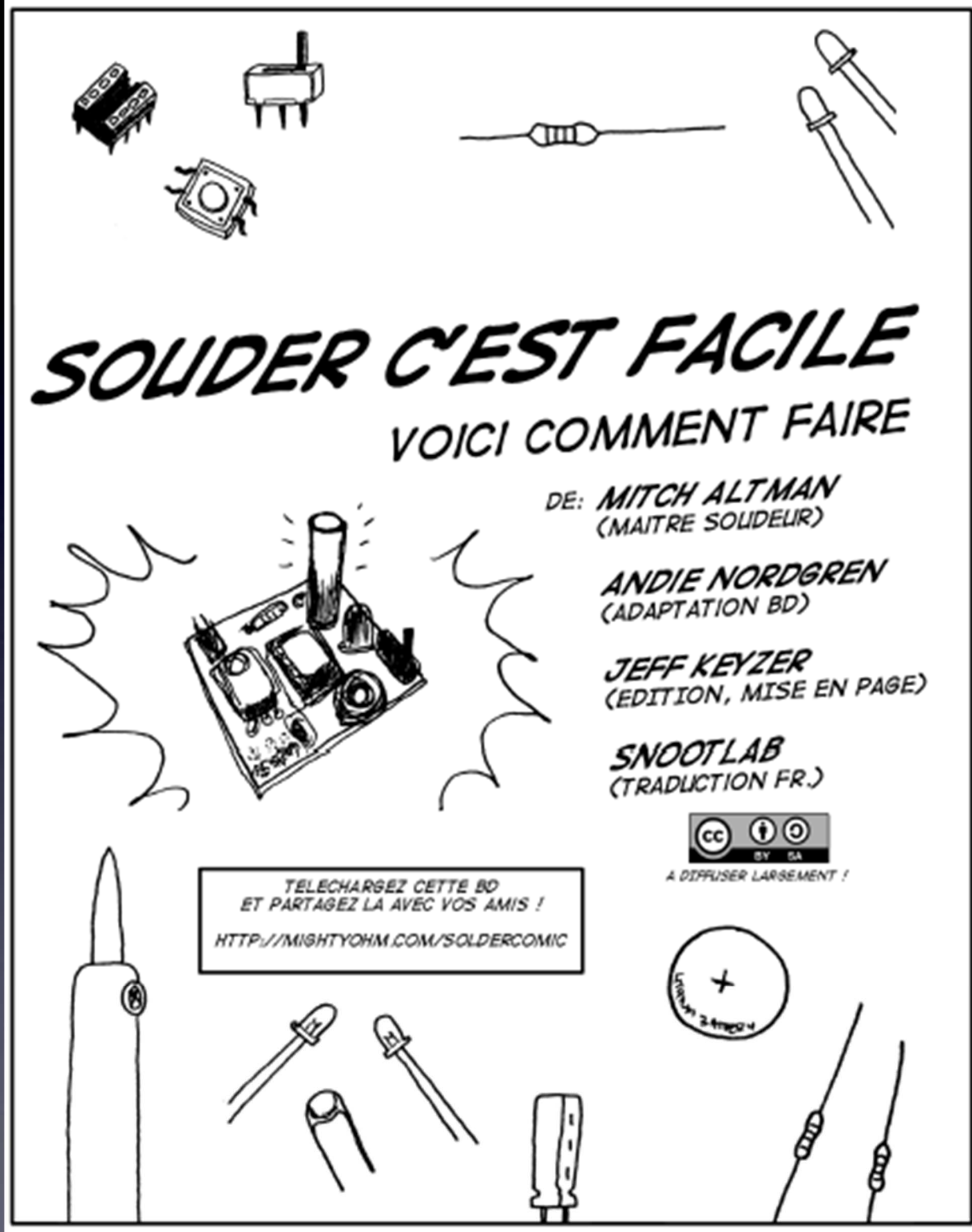
download for free at:
<http://mightyohm.com/soldercomic>
(In many different languages.)

Learn To Solder



download for free at:
<http://mightyohm.com/soldercomic>
(In many different languages.)

Learn To Solder



download for free at:
<http://mightyohm.com/soldercomic>
(In many different languages.)

Learn To Solder

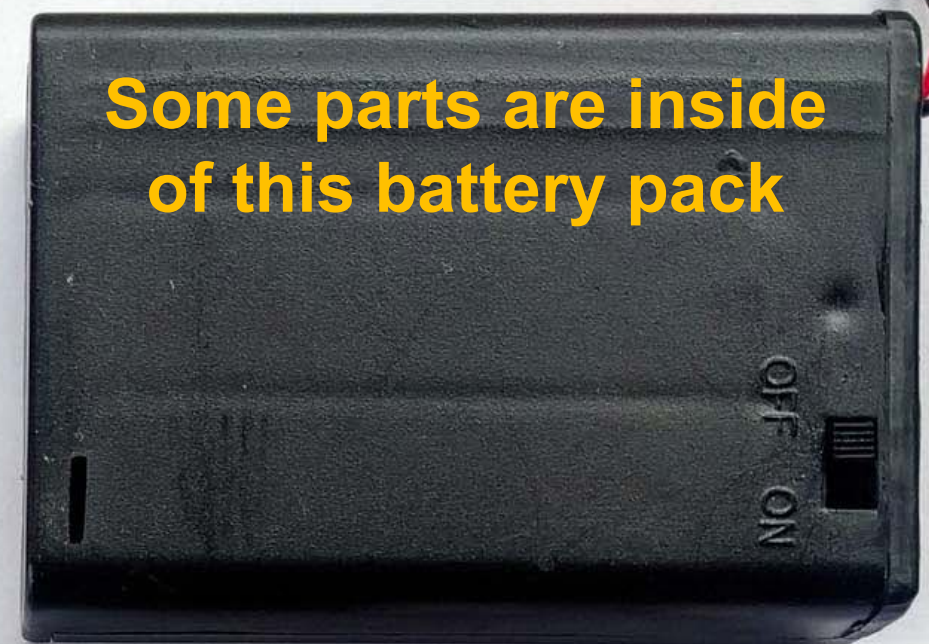
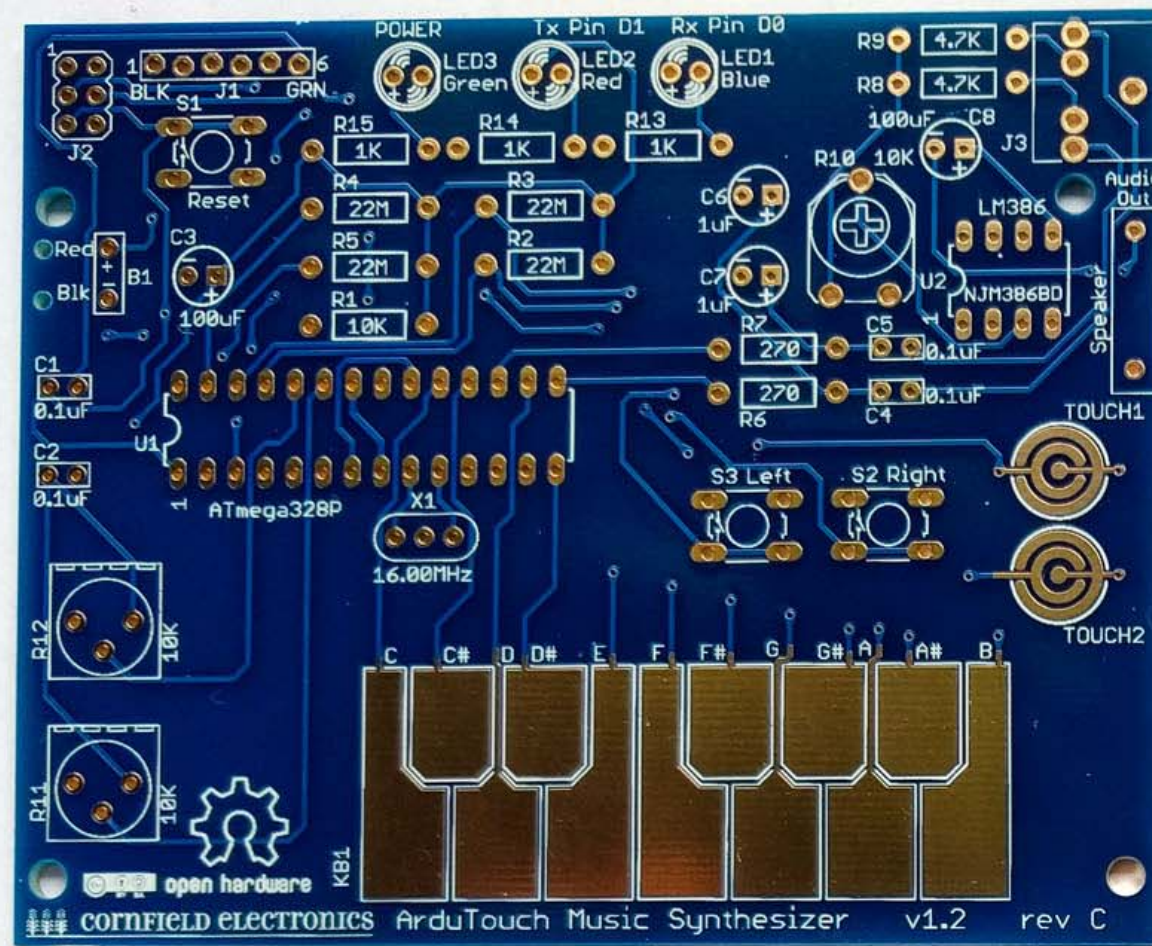


download for free at:
<http://mightyohm.com/soldercomic>
(In many different languages.)

Learn To Solder



download for free at:
<http://mightyohm.com/soldercomic>
(In many different languages.)



Some parts are inside of this battery pack

B1

ATmega328P chip

LM386 chip



U1

U2



Socket (note polarity)

R6, R7 (270) Red, Violet, Brown, Gold



R13, R14, R15 (1K) Brown, Black, Red, Gold



R8, R9 (4.7K) Yellow, Violet, Red, Gold



R1 (10K) Brown, Black, Orange, Gold



R2, R3, R4, R5 (22M) Red, Red, Blue, Gold



Speaker

S1



S2



S3



J1



J3



J2

(not included)



R11



R12



R10



C3



C6



C7



X1



C8



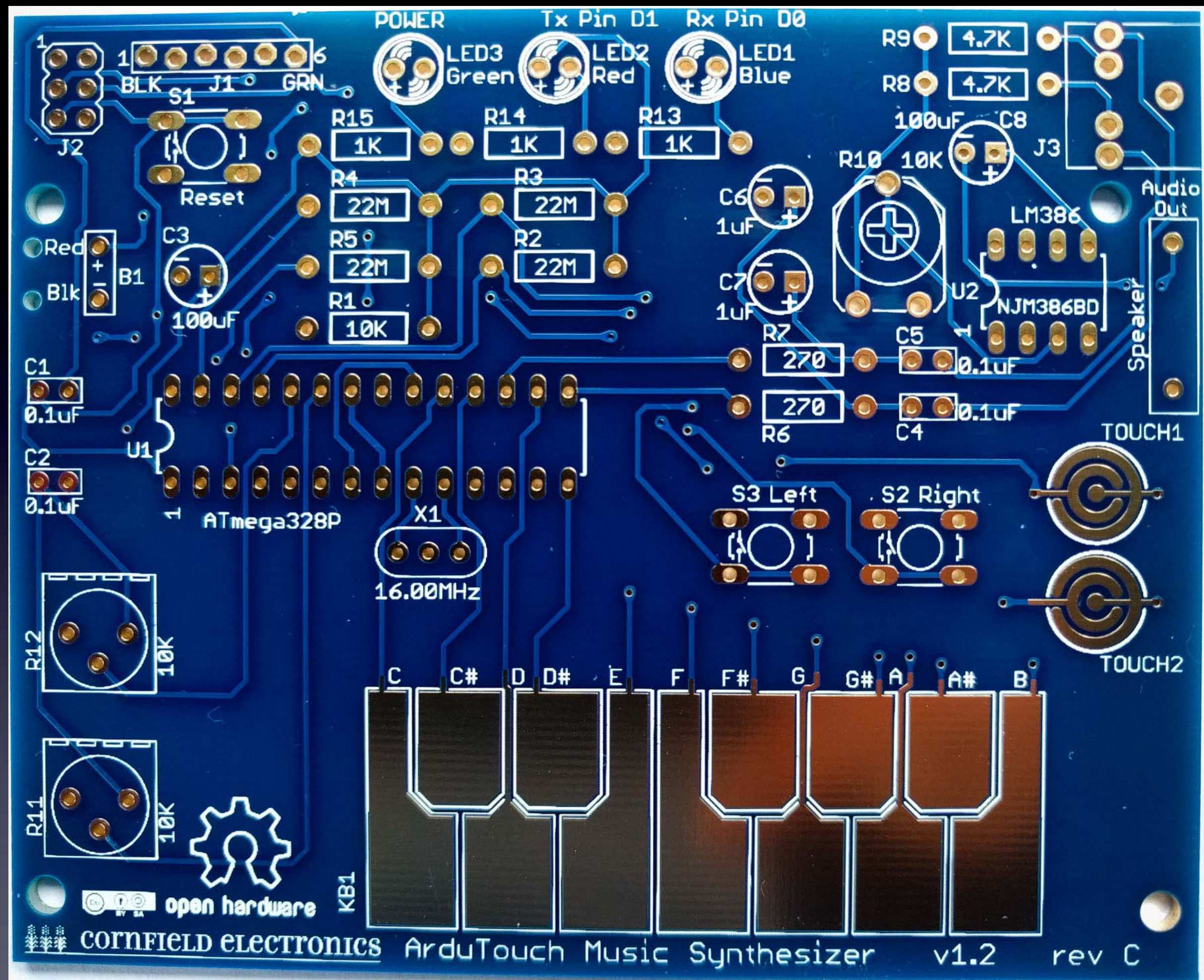
C1, C2, C4, C5



LED3, LED2, LED1



All of the parts



The board we'll solder the parts to



Note:
Since we will use
Lead-Free solder
it is helpful
to also have
flux paste in a syringe
And Isopropyl Alcohol



The tools you'll need:

- soldering Iron (35W or less)
- solder (*more details coming*)
- soldering iron stand
- cellulose kitchen sponge (*not plastic!*)
- *small* wire cutter

Our first part



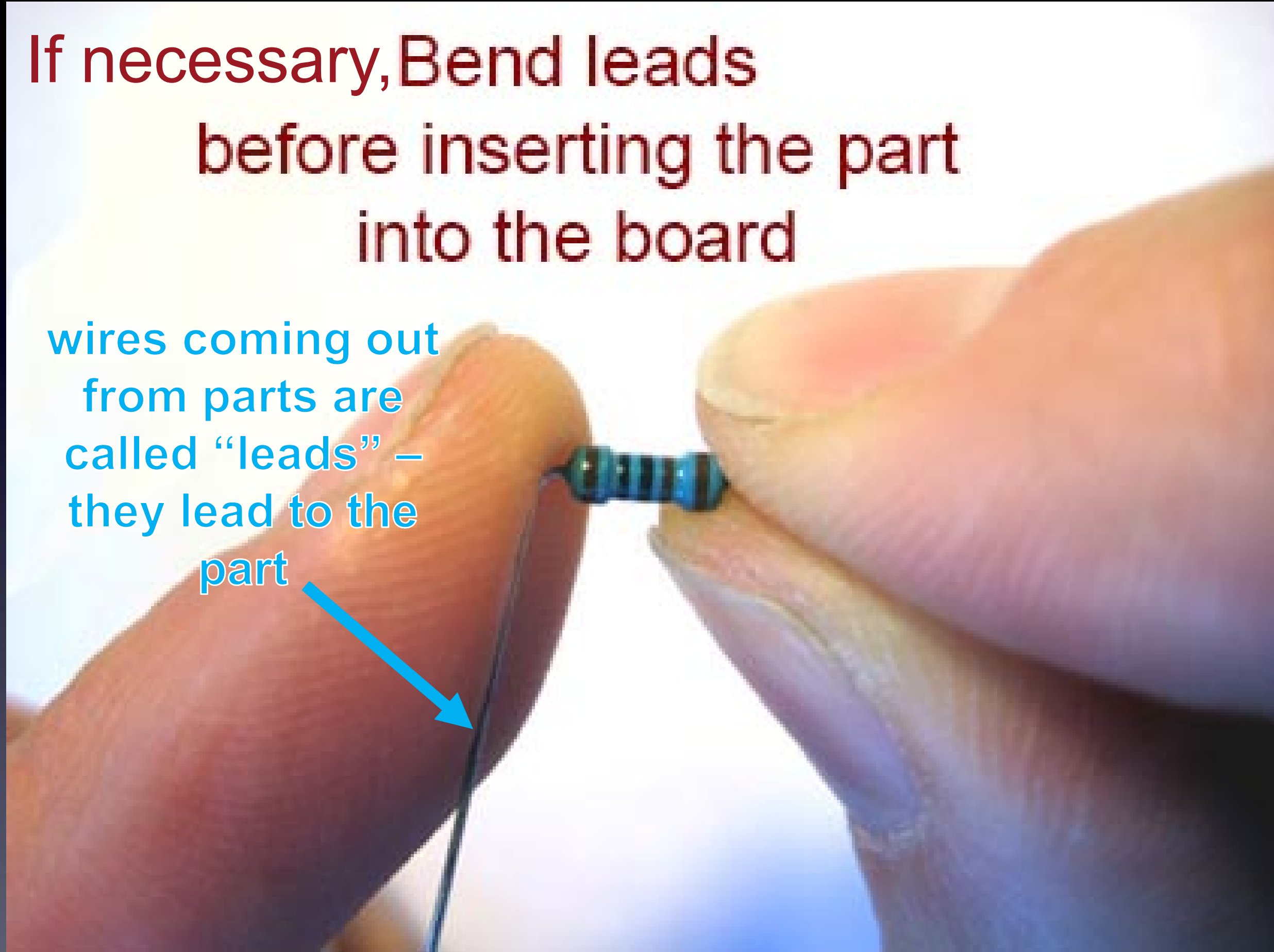
R1: Brown, Black, Orange

(not Brown, Black, ~~Red~~)

Some parts, such as resistors, need their leads bent first

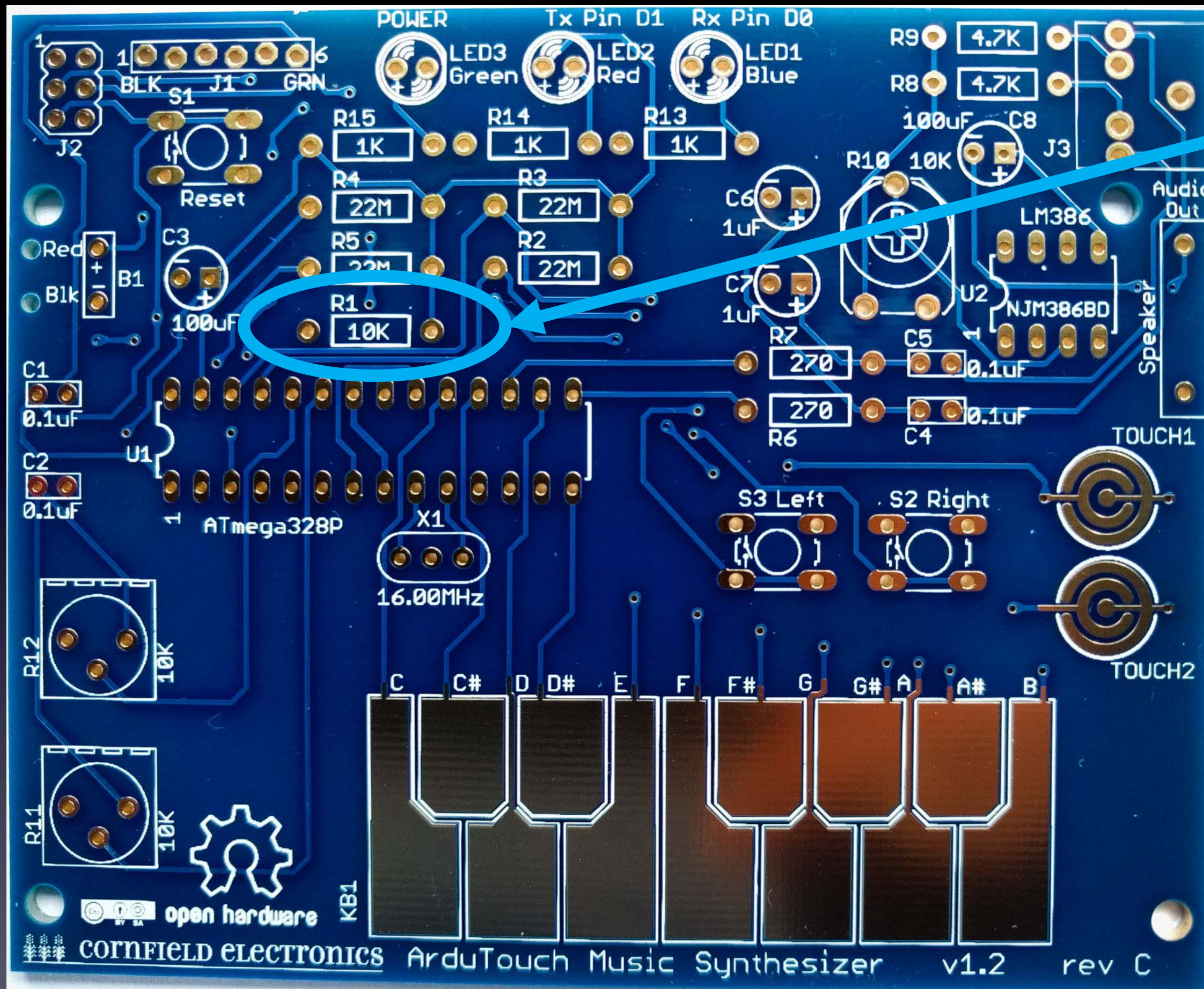
If necessary, Bend leads
before inserting the part
into the board

wires coming out
from parts are
called “leads” –
they lead to the
part





R1 – this is how it will look *before* inserting it into the board



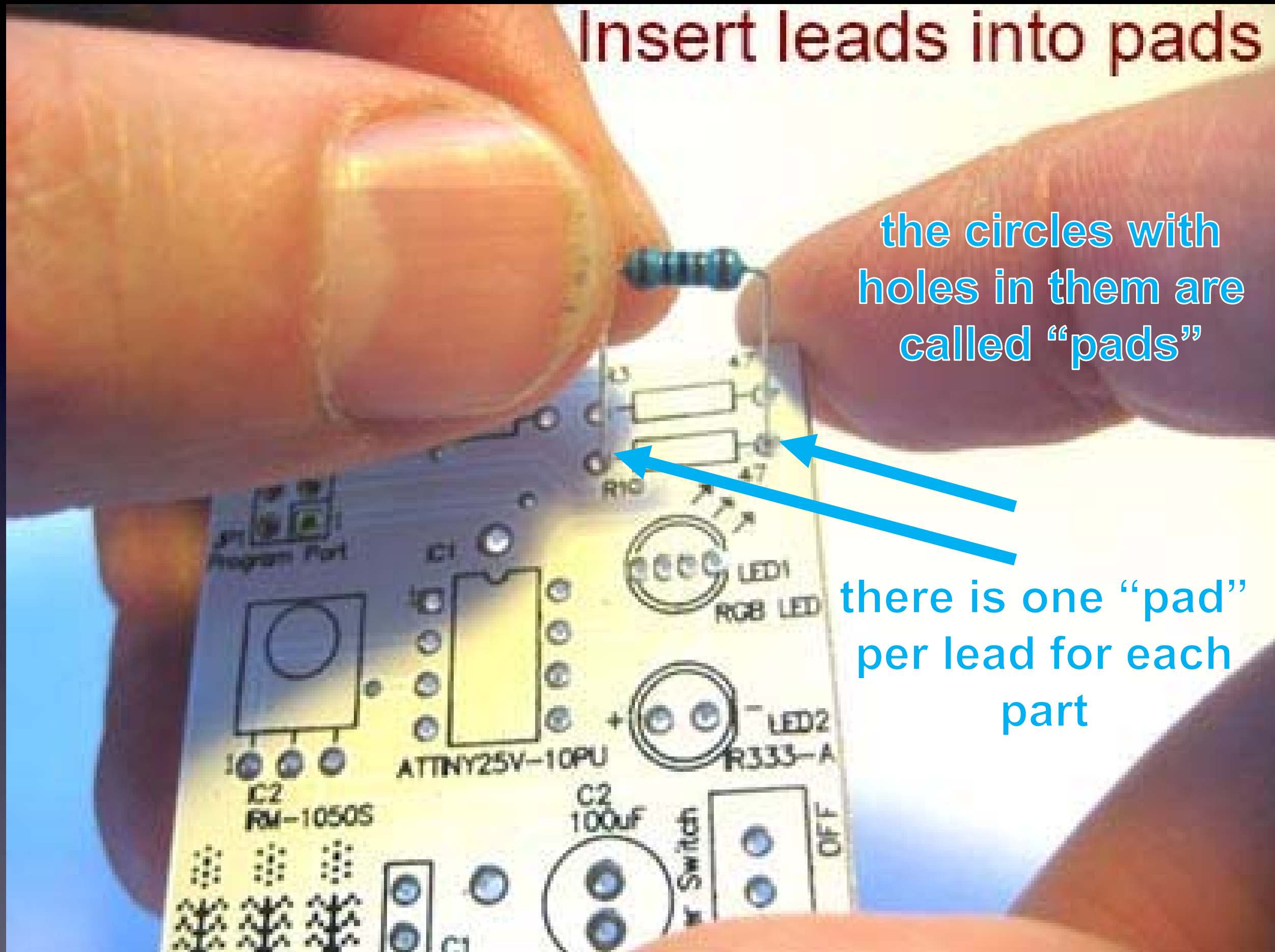
R1

R1 – this is where it goes

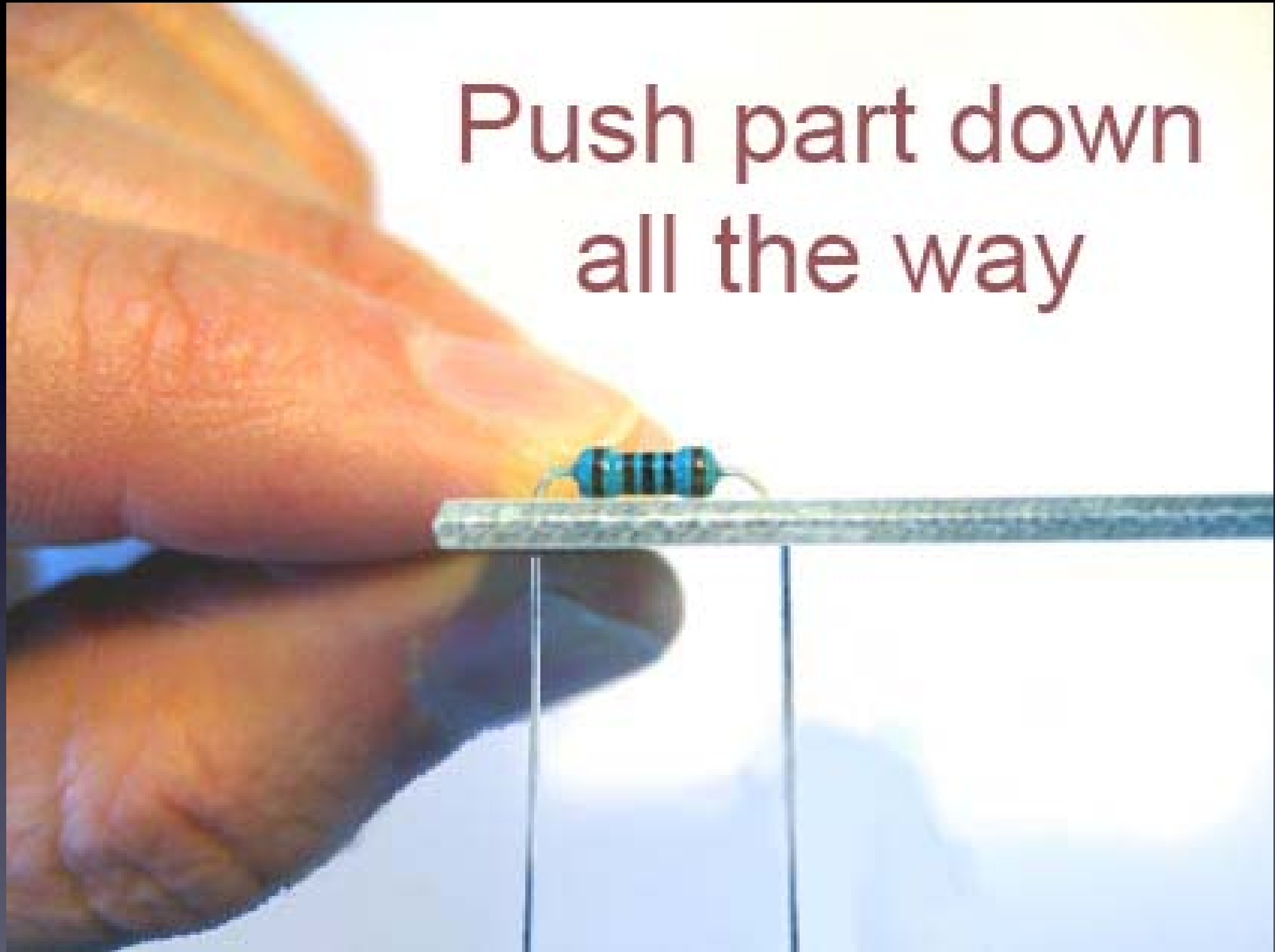
Insert leads into pads

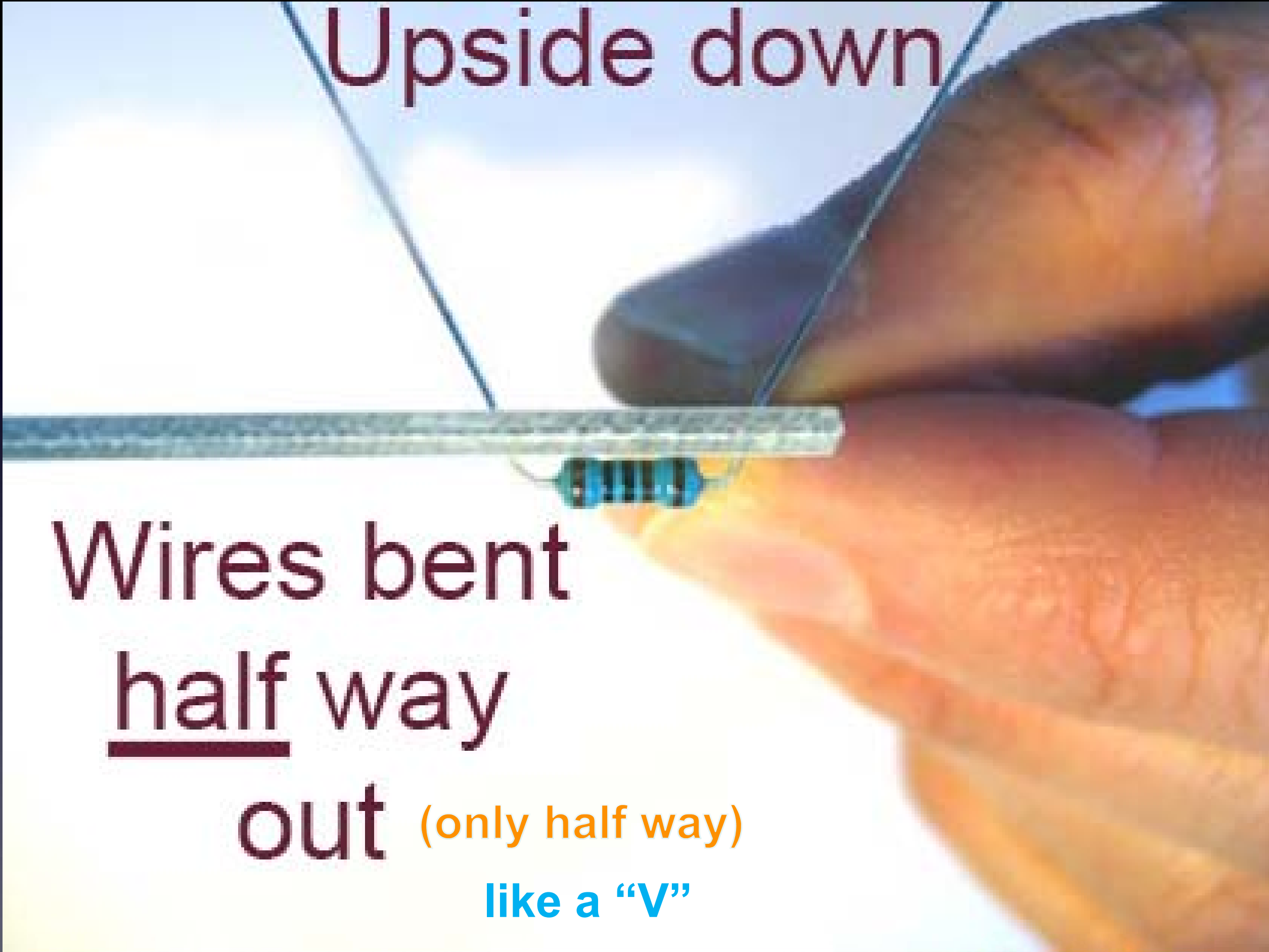
the circles with
holes in them are
called "pads"

there is one "pad"
per lead for each
part



Push part down
all the way

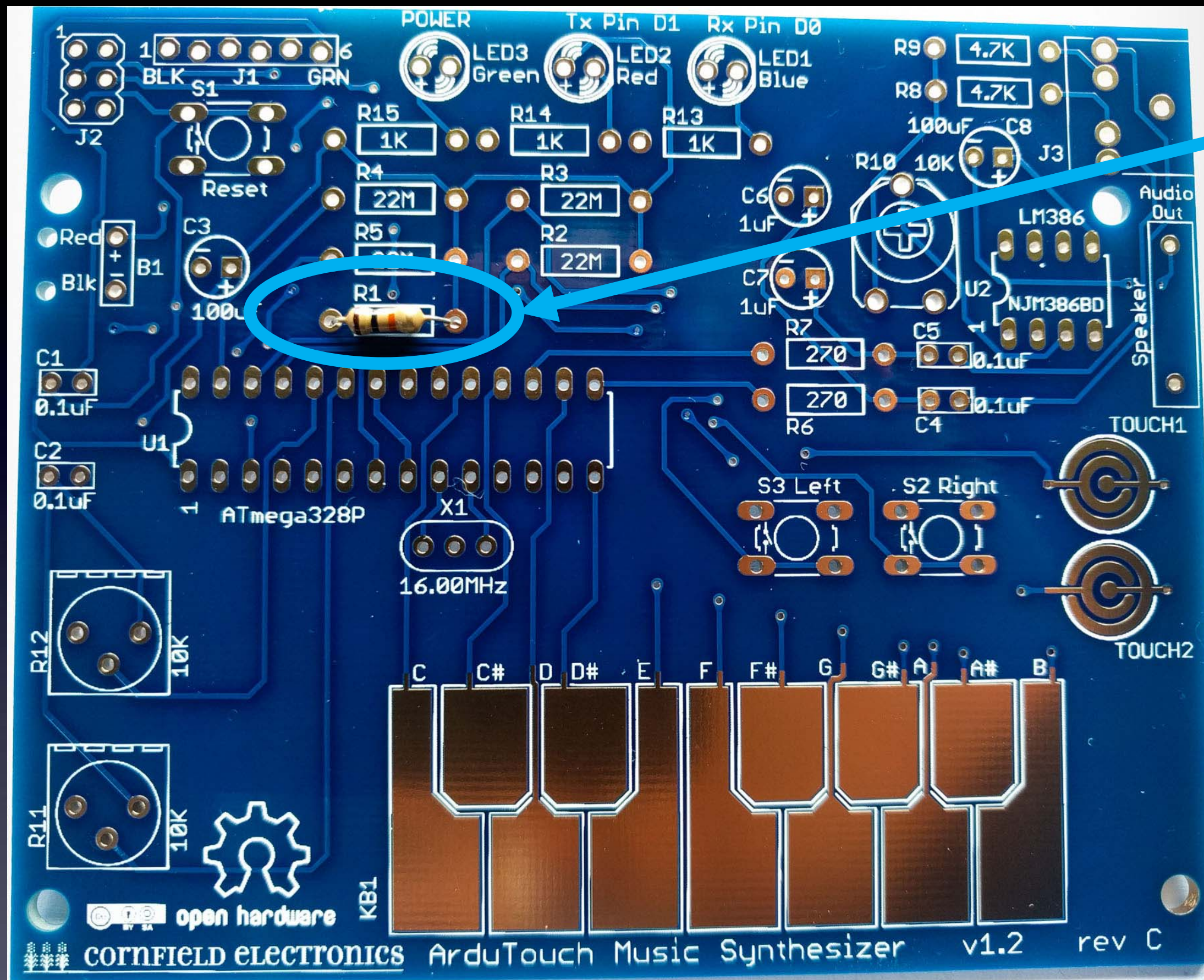


A close-up photograph of a person's hand holding a thin metal wire. A small, cylindrical electronic component, likely a resistor, is attached to the wire. The wire is bent into a V-shape at the point where the component is attached. The background is a plain, light-colored surface.

Upside down

Wires bent
half way
out (only half way)
like a “V”

so that the part won't fall out while soldering it



R1 – inserted into the board



How to hold a soldering iron

(Like a pencil – held from underneath)

Important

The best kind of solder for DIY electronics:

(Sn – Tin / Pb – Lead)

63/37 rosin core,

0.031" (0.8mm) diameter (or smaller)

(60/40 is also good)

Note:

Most

***Lead-Free* solder**

has poisonous fumes!

This is what we will use:

A good kind of solder for DIY
electronics:

*This is the only good **Lead-Free** solder I have found!*
(after years of searching)



Kester
K100LD Rosin
(not "No Clean")
0.031" diameter (0.8mm)

This is what we will use:

A good kind of solder for DIY electronics:

*This is the only good **Lead-Free** solder I have found!*
(after years of searching)



Kester **K100LD Rosin** Solder

0.031" diameter (0.8mm)

Note:

Since we will use **Lead-Free** solder
it is *helpful*
to also have
flux paste in a syringe
And Isopropyl Alcohol



99%



Another good kind of solder for DIY electronics:

*This is another good **Lead-Free** solder I have found!*



**Duratool
D01685 Rosin**

0.7mm diameter

*(as good as the
Kester K100LD Rosin)*

Another good kind of solder for DIY electronics:

*This is another good **Lead-Free** solder I have found!*



**MG Chemicals
4900 Rosin (112g, 227g, 454g)**

0.8mm diameter

*(as good as the
Kester K100LD Rosin)*

3 Safety Tips...

Safety Tip #1:

Hot !!

(When you touch the tip,
you will let go quickly every time!)

Safety Tip #2:

Soldering chemicals
are toxic

But they easily wash off your hands
with soap and water

Safety Tip #3:

(coming soon)

2 secrets
to good soldering...

Secret #1:

Clean the tip!

(before every solder connection)

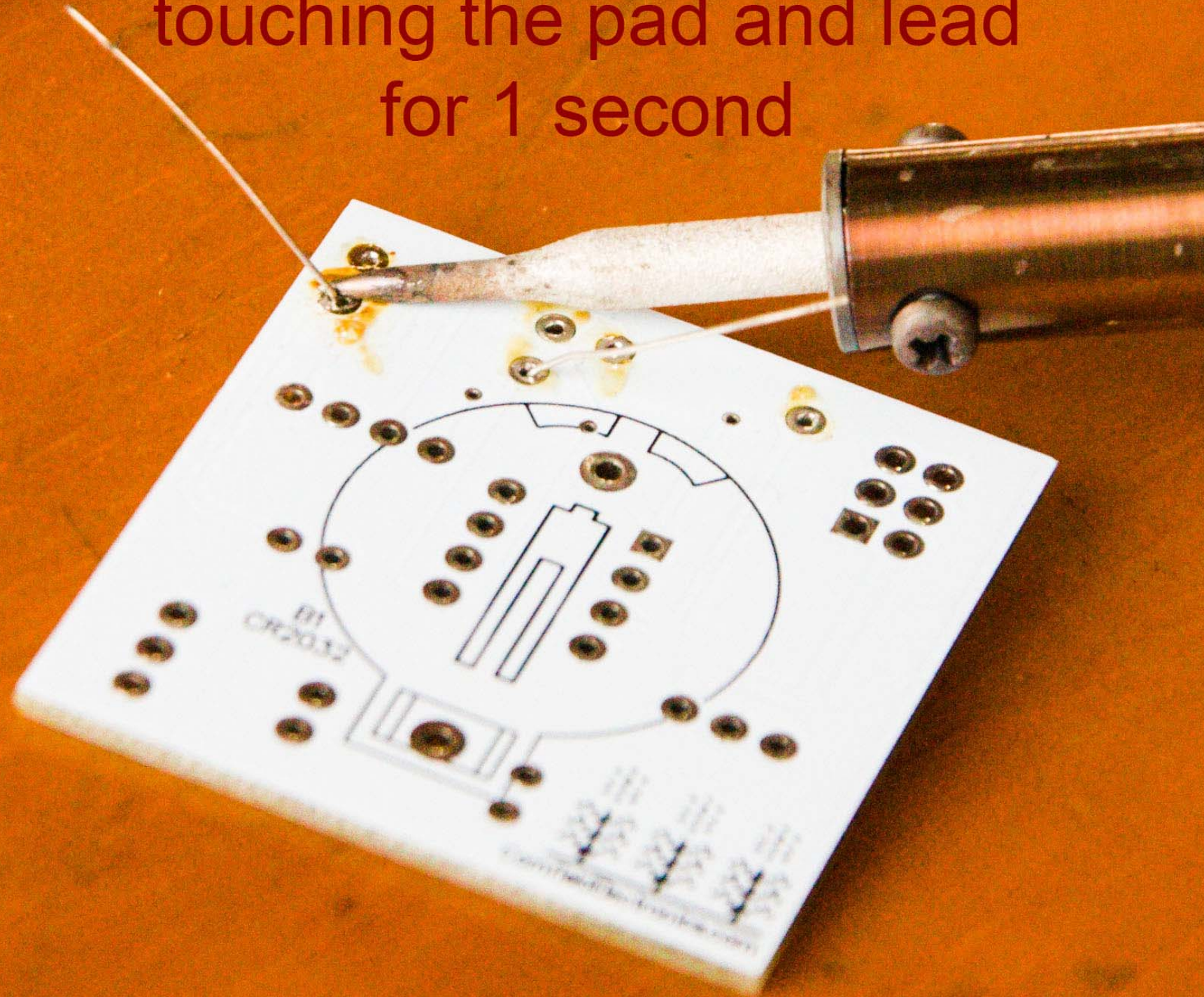
Bang (lightly) 3 times,

Swipe, Rotate, Swipe (on the sponge):

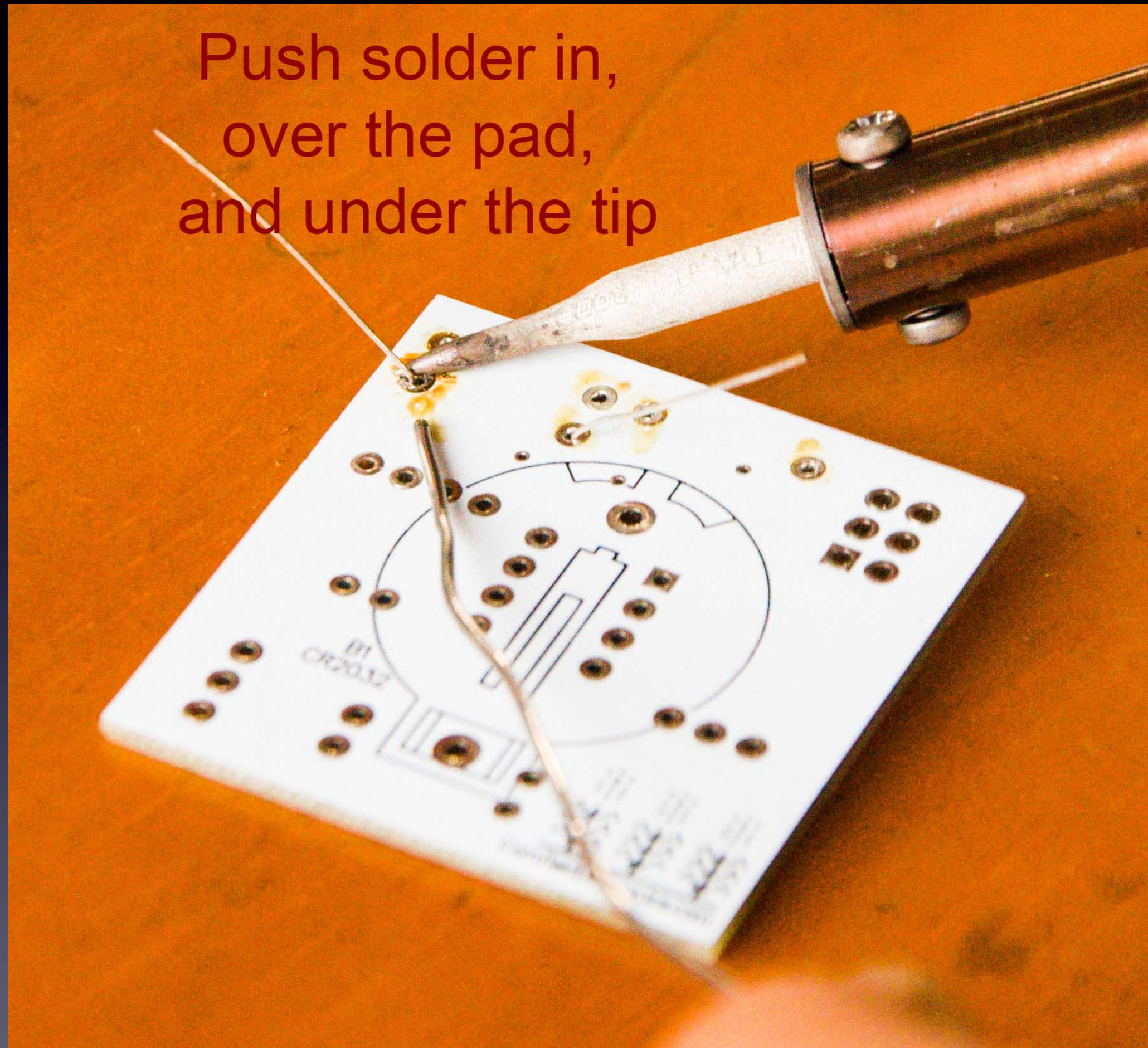
Keep the tip shiny silver!

knock solder off the tip

Lay clean tip across half of the pad,
touching the pad and lead
for 1 second

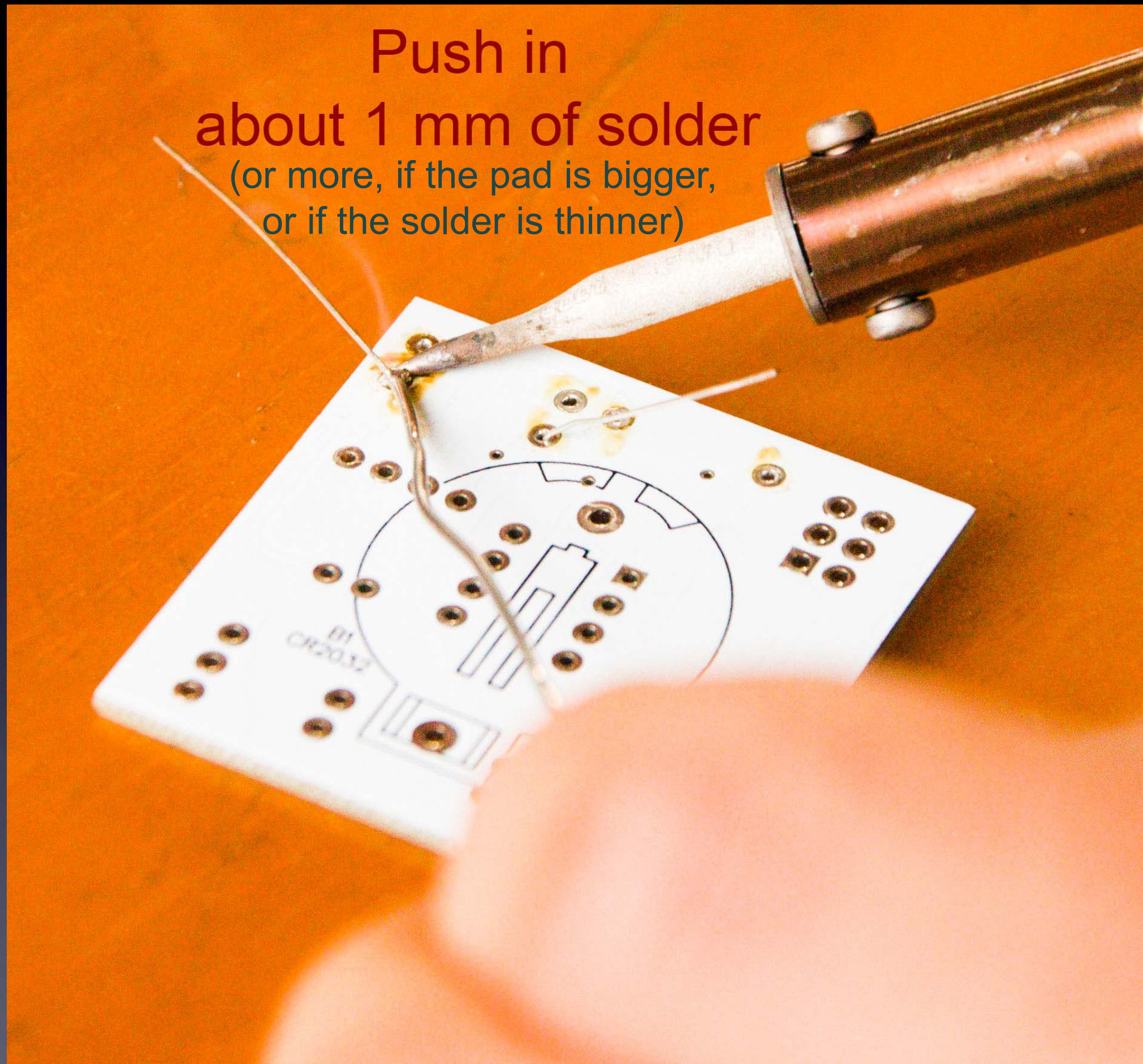


Do this quickly (slowly doesn't work well) – solder in & out in about 1 second



Make sure solder melts on the underside of the soldering iron tip
(not the side or top of the soldering iron tip)!

Do this quickly (slowly doesn't work well) – solder in & out in about 1 second



Make sure solder melts on the underside of the soldering iron tip
(not the side or top of the soldering iron tip)!



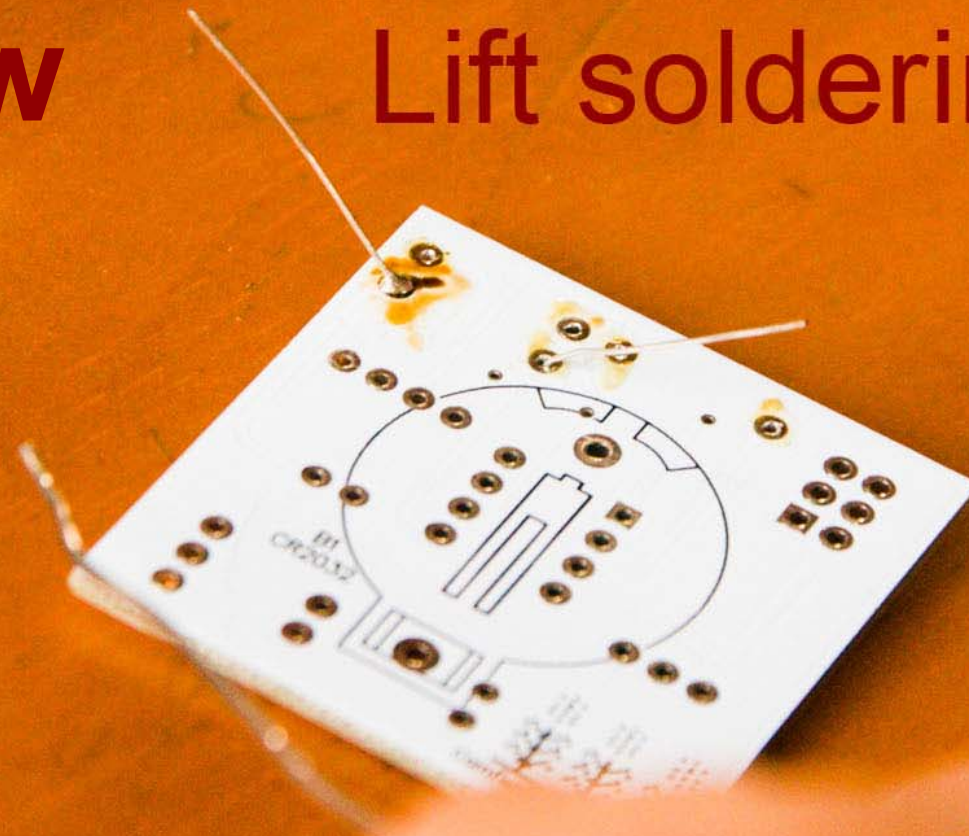
Pull solder away,
But keep holding soldering iron down
for 1 more second !!

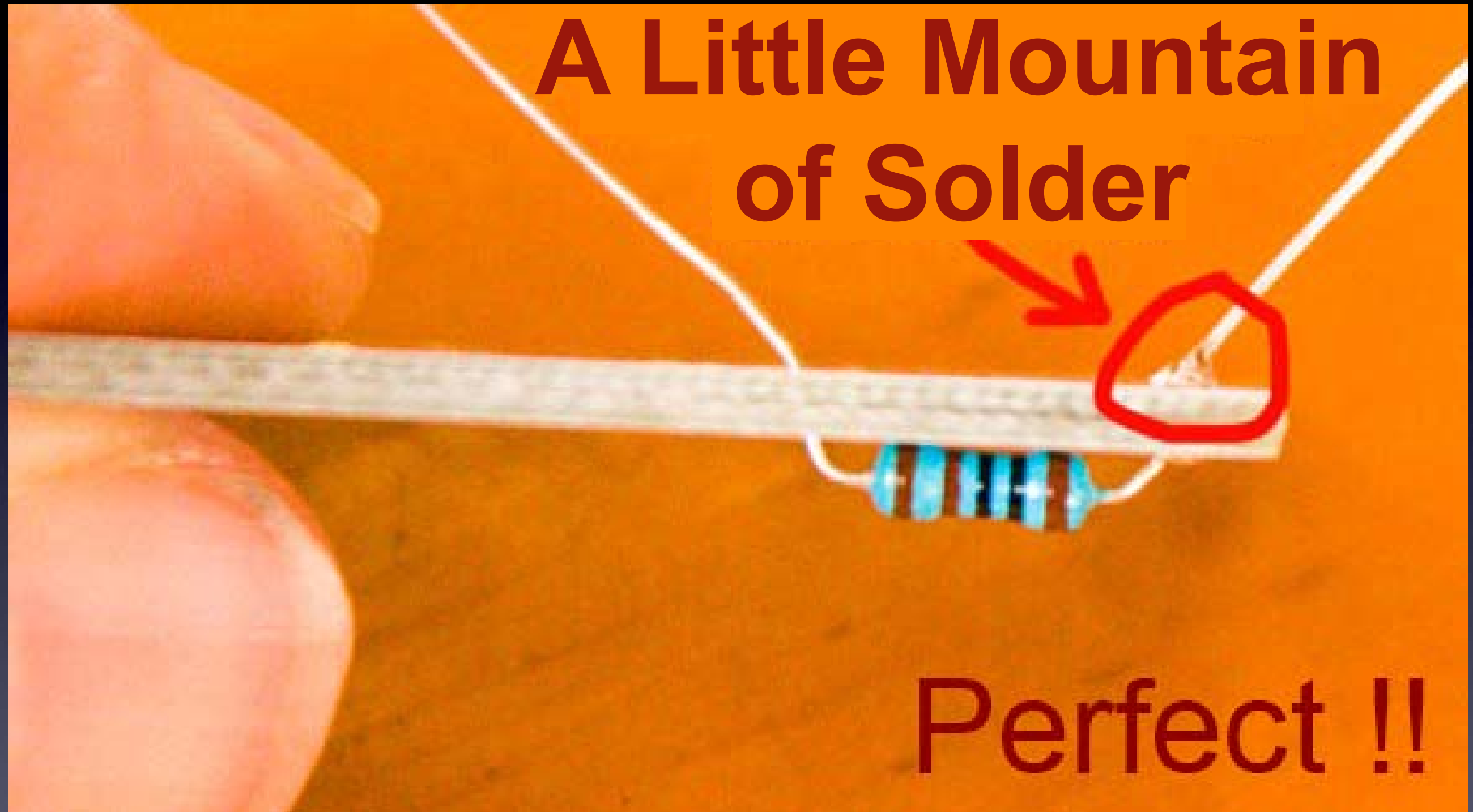
Secret #2:

Keep hot tip down
1 second
for solder to flow !!

Now

Lift soldering iron





If you can see any of the pad, or the hole, you need more solder
– so, just do all the steps again to make it perfect.

The Rhythm !

is just as important as the preceding steps!

The Rhythm !

and speed (about 1 second per step)



The Rhythm !

and speed (about 1 second per step)

Clean the tip



The Rhythm !
and speed (about 1 second per step)



Tip **Down**

The Rhythm !
and speed (about 1 second per step)



Solder **In**

The Rhythm !
and speed (about 1 second per step)



Solder **Out**

The Rhythm !
and speed (about 1 second per step)



WAIT !

The Rhythm !
and speed (about 1 second per step)



Lift Tip

The Rhythm !

and speed (about 1 second per step)



The Rhythm !

and speed (about 1 second per step)

Clean the tip



The Rhythm !
and speed (about 1 second per step)



Tip **Down**

The Rhythm !
and speed (about 1 second per step)



Solder **In**

The Rhythm !
and speed (about 1 second per step)



Solder **Out**

The Rhythm !
and speed (about 1 second per step)



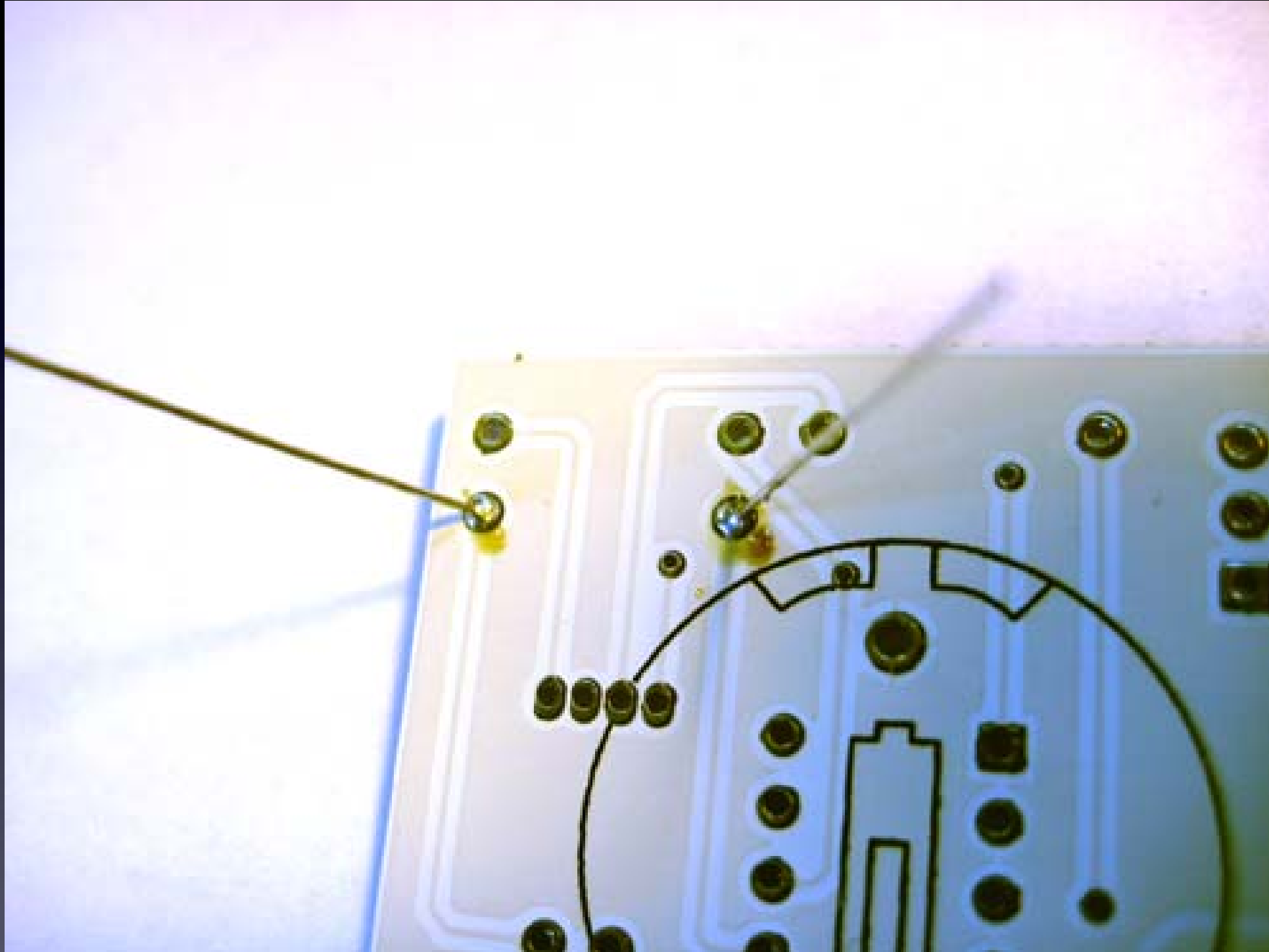
WAIT !

The Rhythm !
and speed (about 1 second per step)



Lift Tip

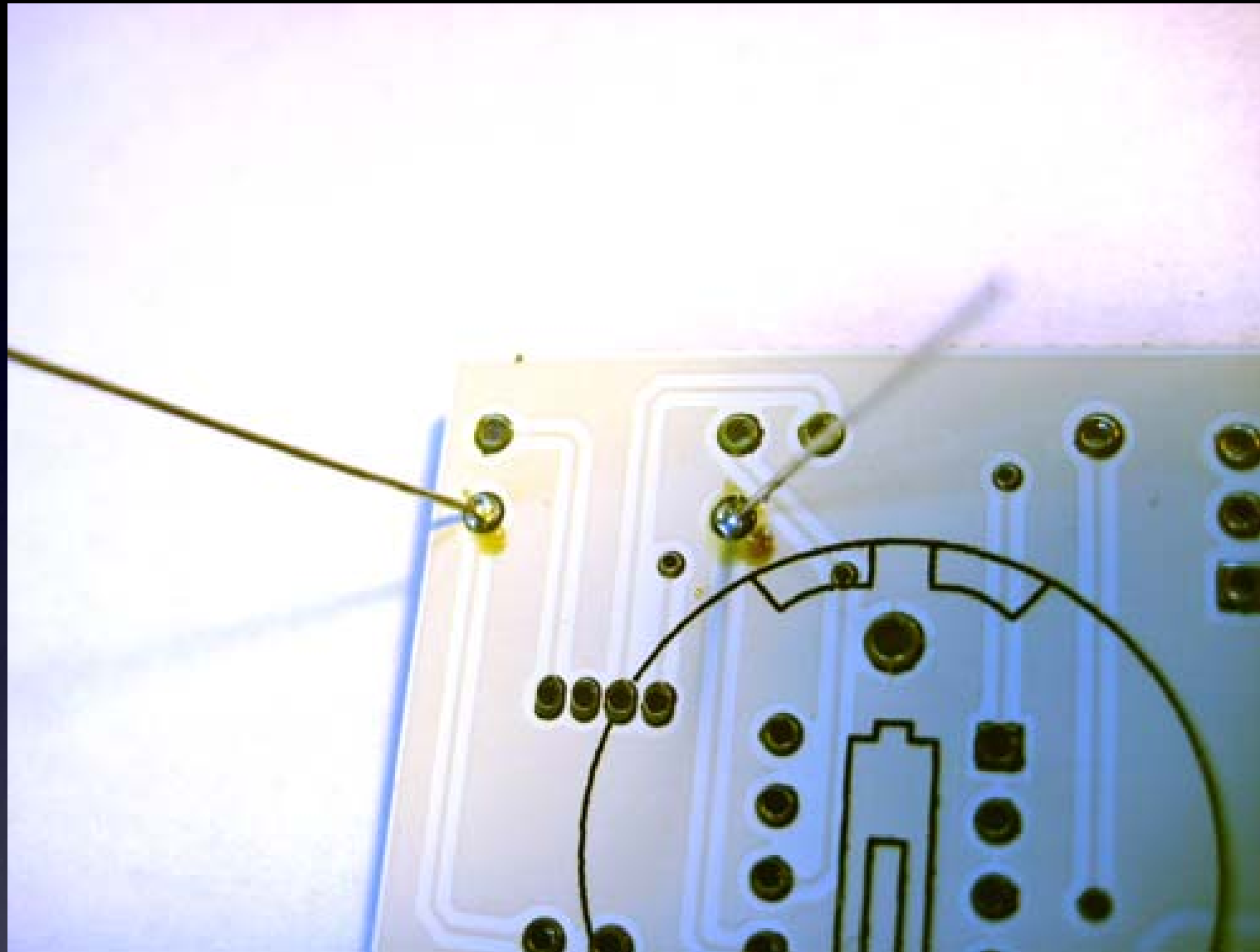
Solder all of the leads of the part to the board



For this part, there are two leads

Here you can see two good solder connections

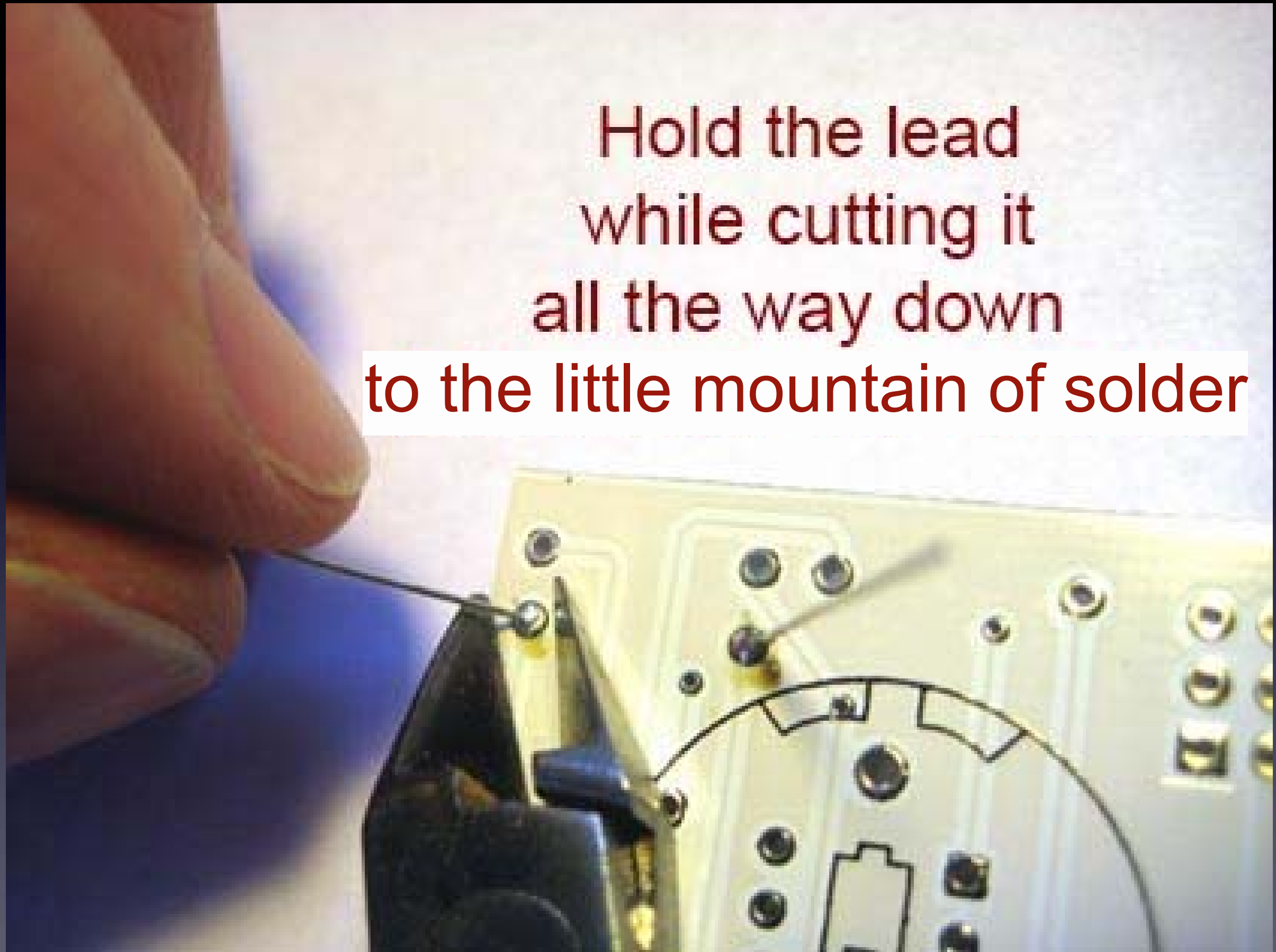
Two good solder connections



- Little mountains (not flat)
- Pads totally covered in solder
- Can't see the hole
- No connections to other pads

Now cut the leads short

Hold the lead
while cutting it
all the way down
to the little mountain of solder



Cutting with the tip of the wire cutter gives you more control

Safety Tip #3:

Hold or cover the lead !

(or it will fly into your eye!)

(They like doing that – so please hold or cover the lead when you cut.)



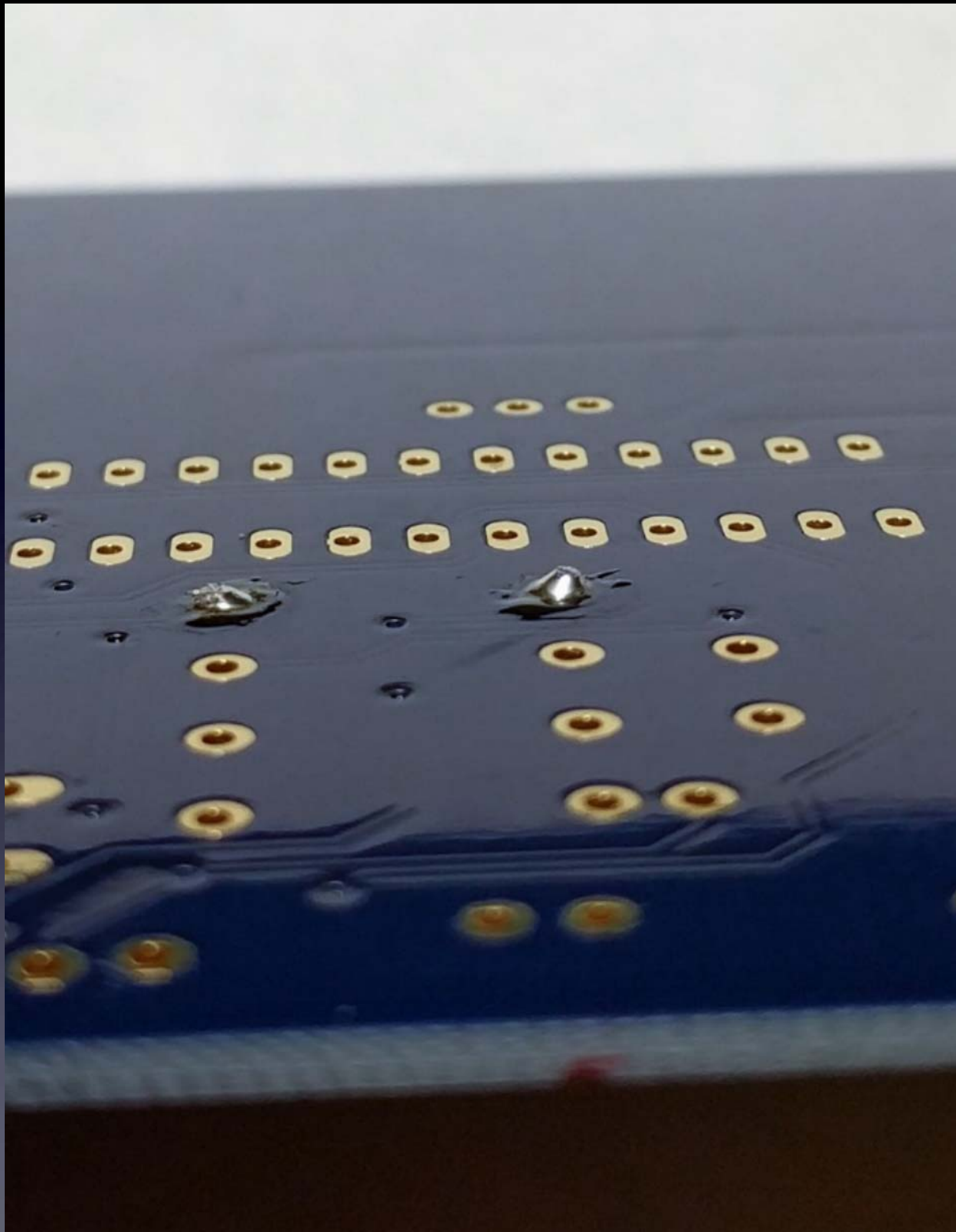
All done !

No wires sticking out

R1 soldered to the board

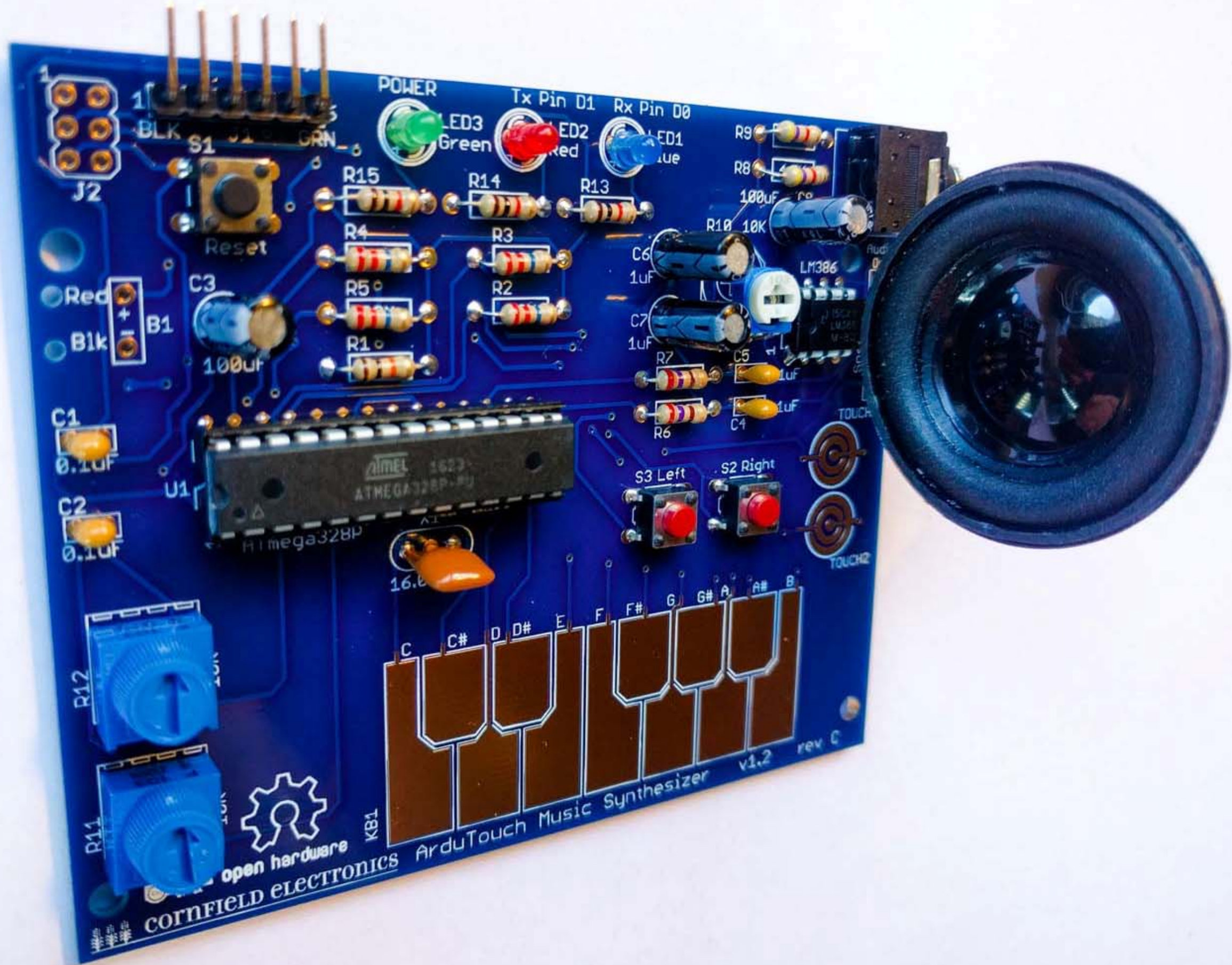
Notice that:

- Each connection is a small mountain (not flat)
- You cannot see any pad (they're totally covered with solder)
- You cannot see the holes (they're totally covered with solder)
- No connections to other pads



One part at a time

Till all the parts are soldered

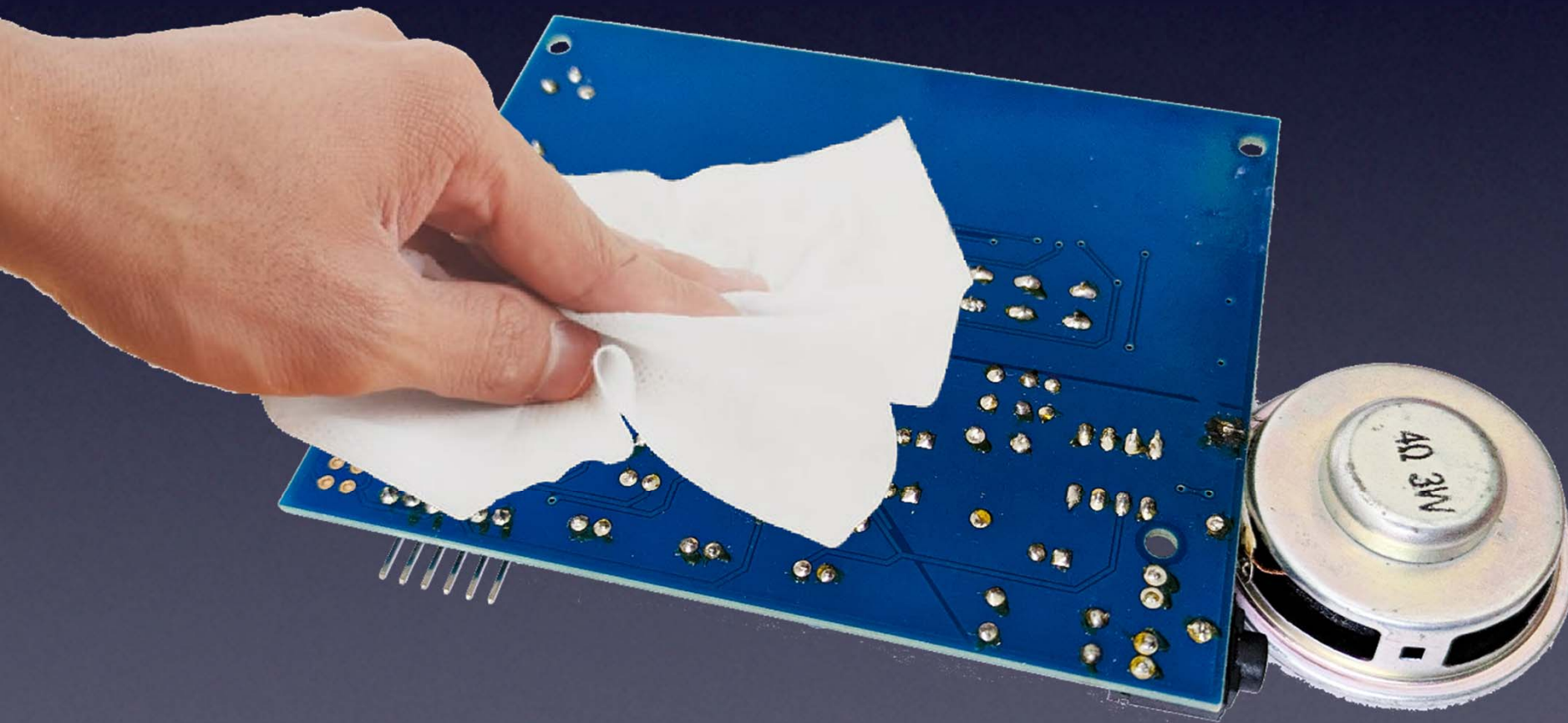


And it will look like this when you're done.

If you used any *flux paste* for re-working problems



The bottom of the PCB will be sticky from the flux



You can clean it with a cloth wet with *Isopropyl Alcohol*

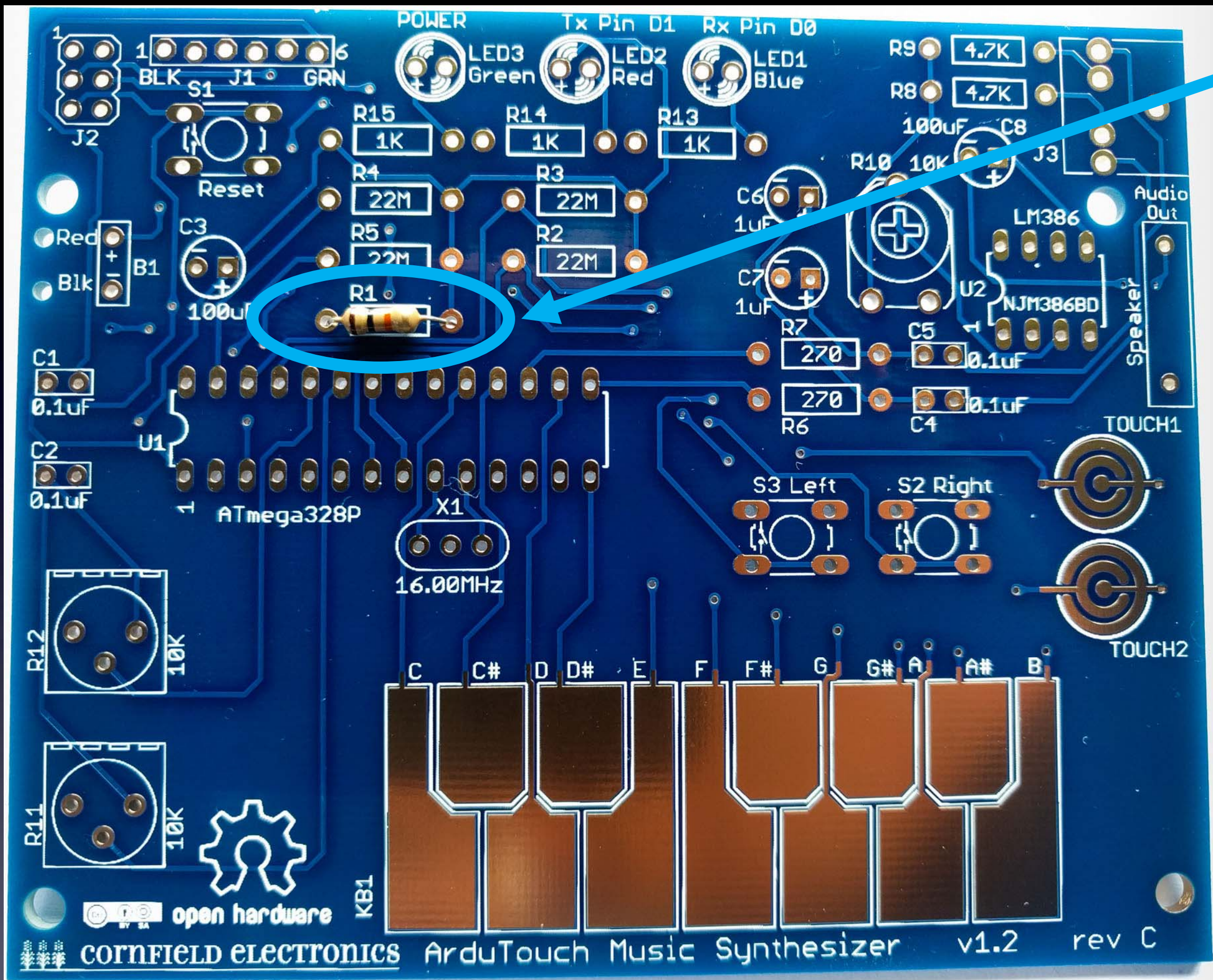
Then put in the batteries,

Turn it on,

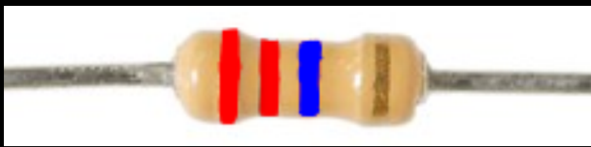
And it works!

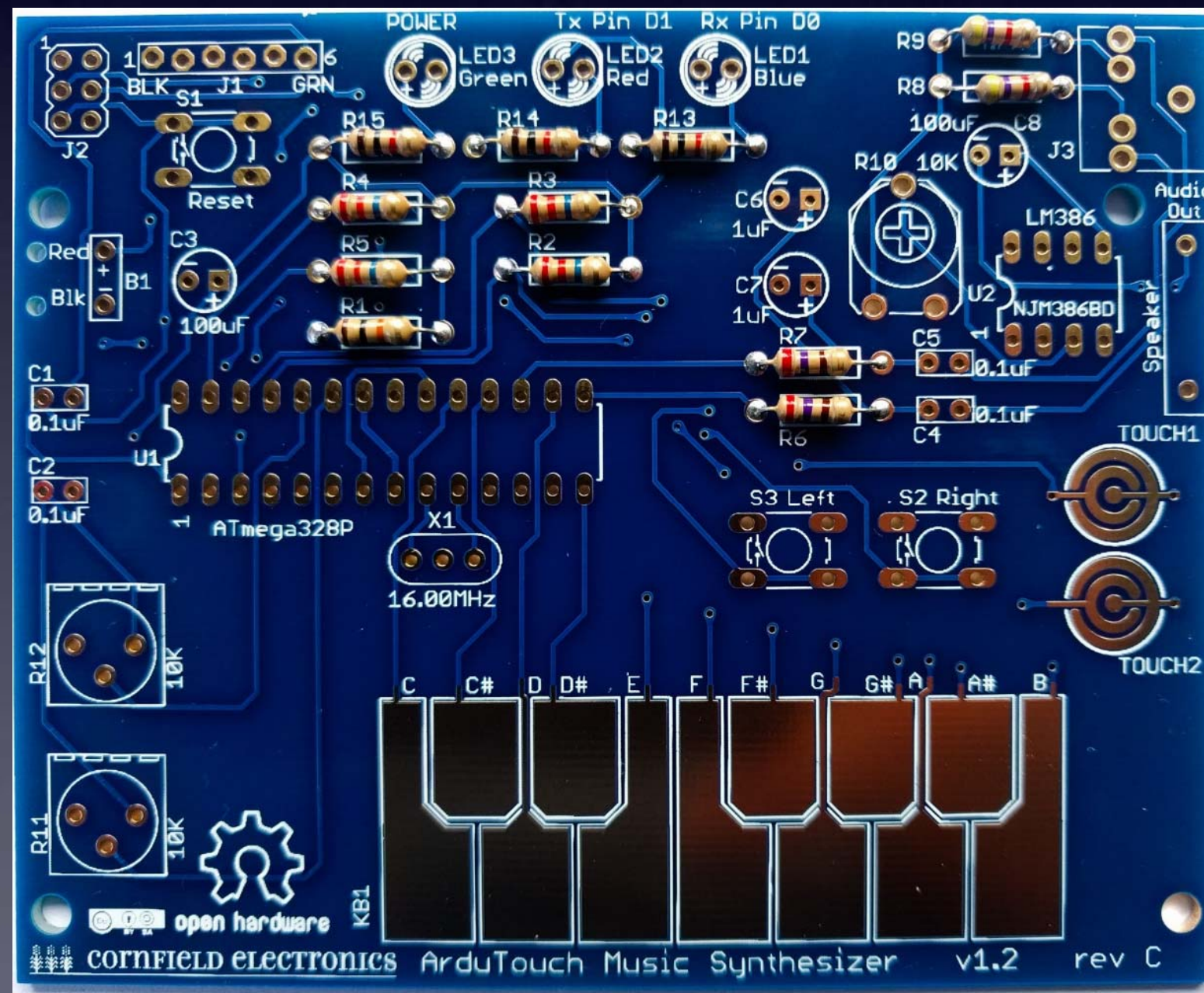
(Or you start debugging.)

Let's start!



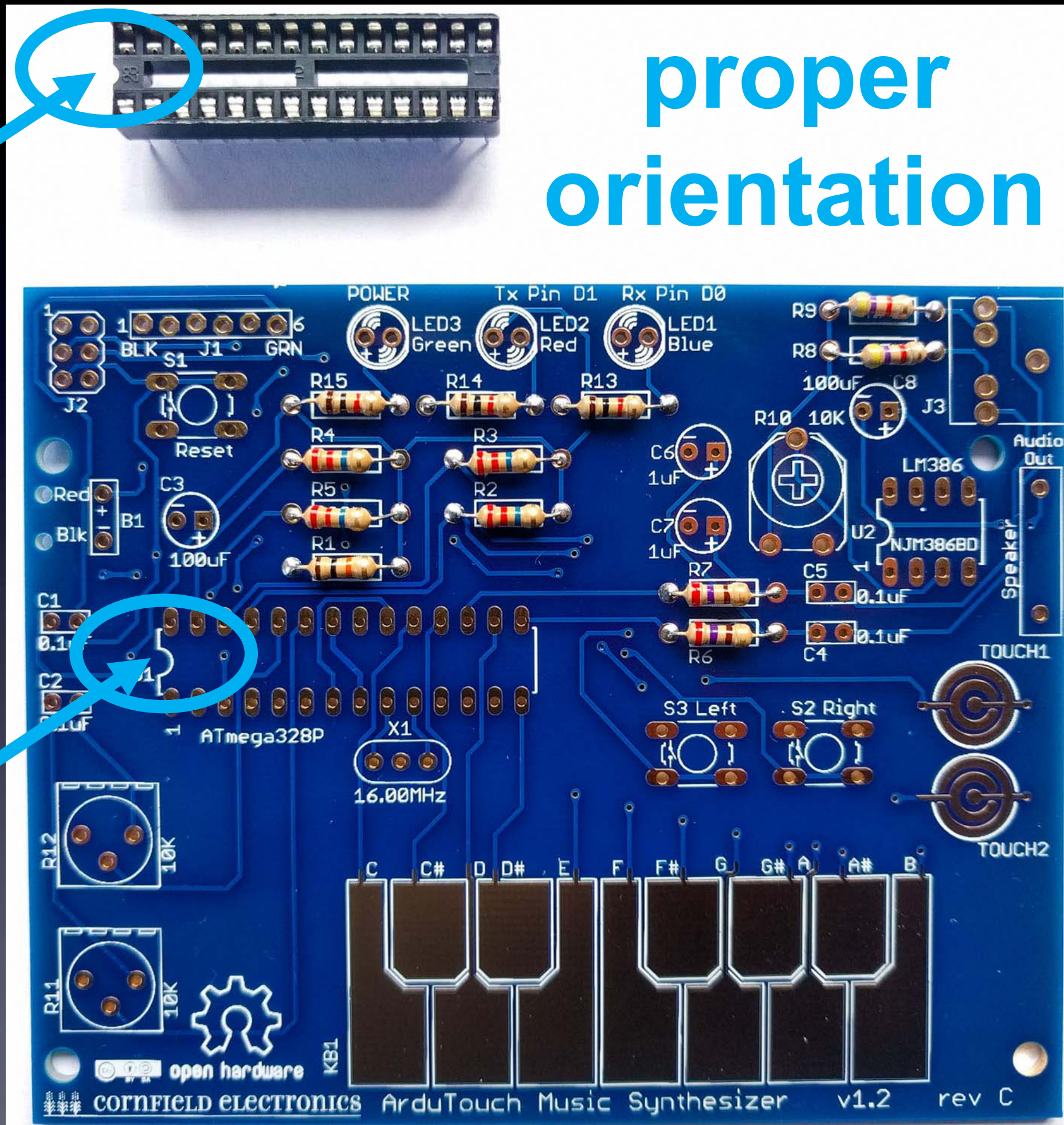
If you haven't done so already, solder R1: brown, black, orange

R1:		10K: Brown, Black, Orange
R2, R3, R4, R5:		22M: Red, Red, Blue
R6, R7:		270: Red, Violet, Brown
R8, R9:		4.7K: Yellow, Violet, Red
R13, R14, R15:		1K: Brown, Black, Red

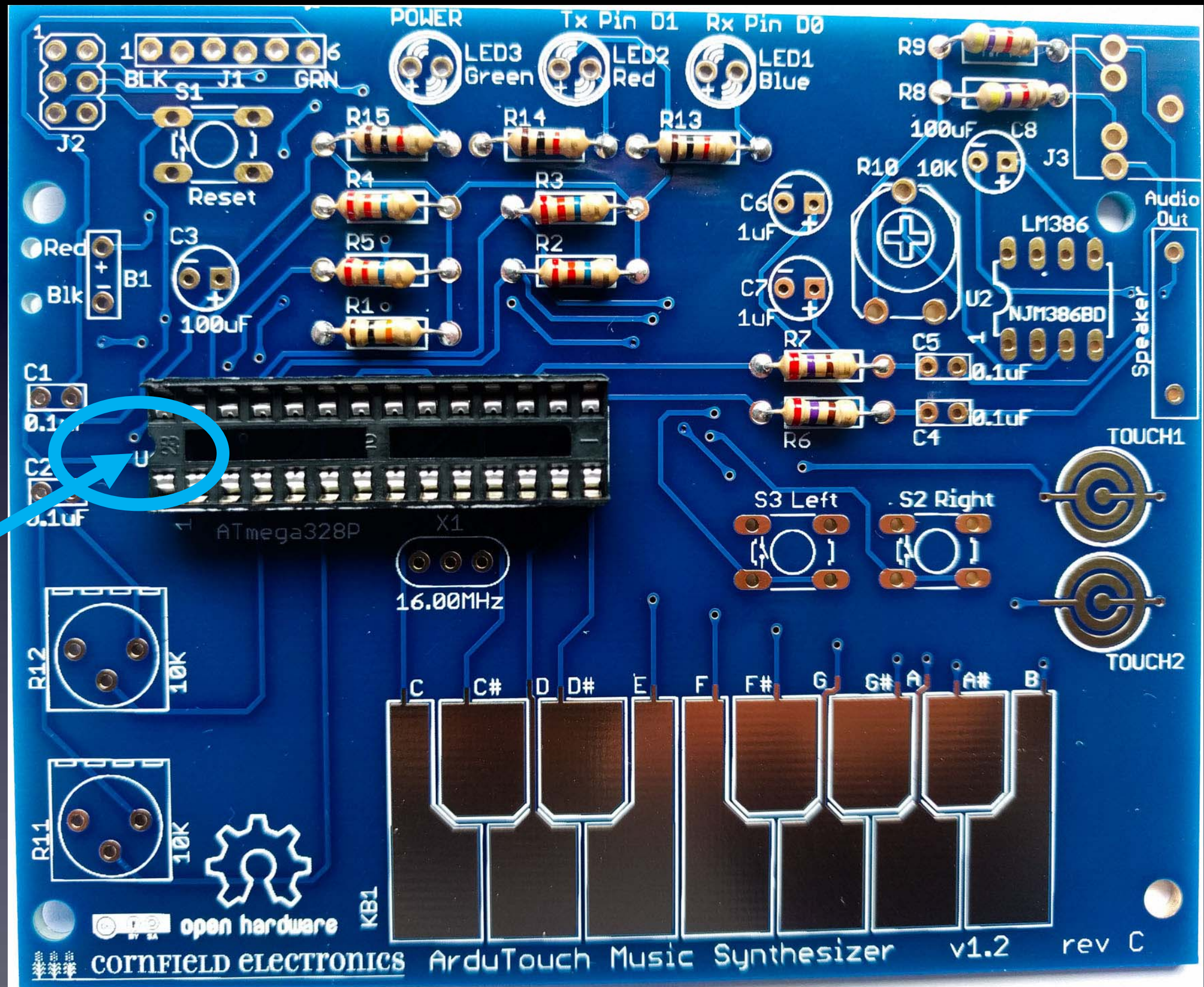


U1: microcontroller socket

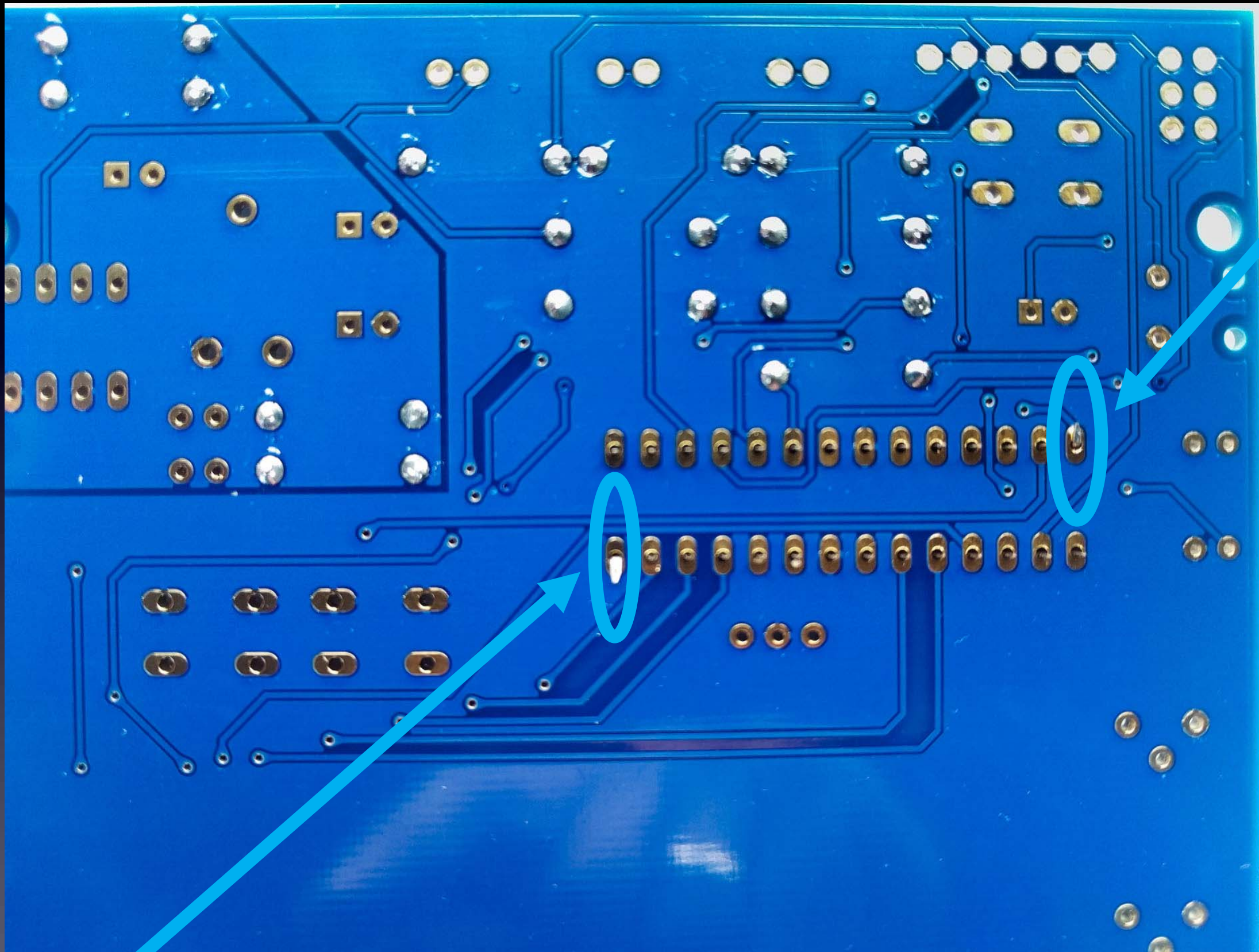
proper
orientation



U1: microcontroller socket: inserted correctly

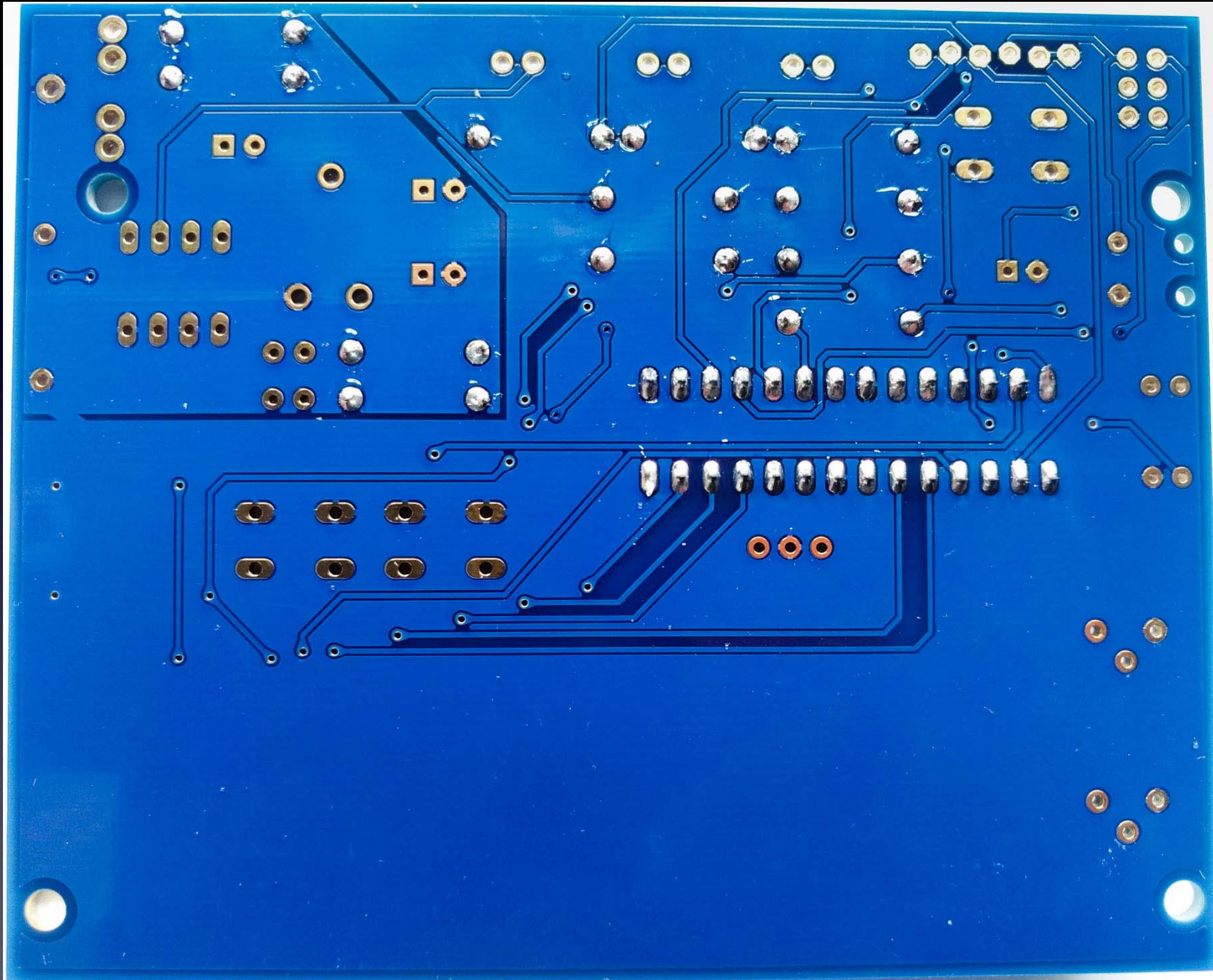


U1: microcontroller socket



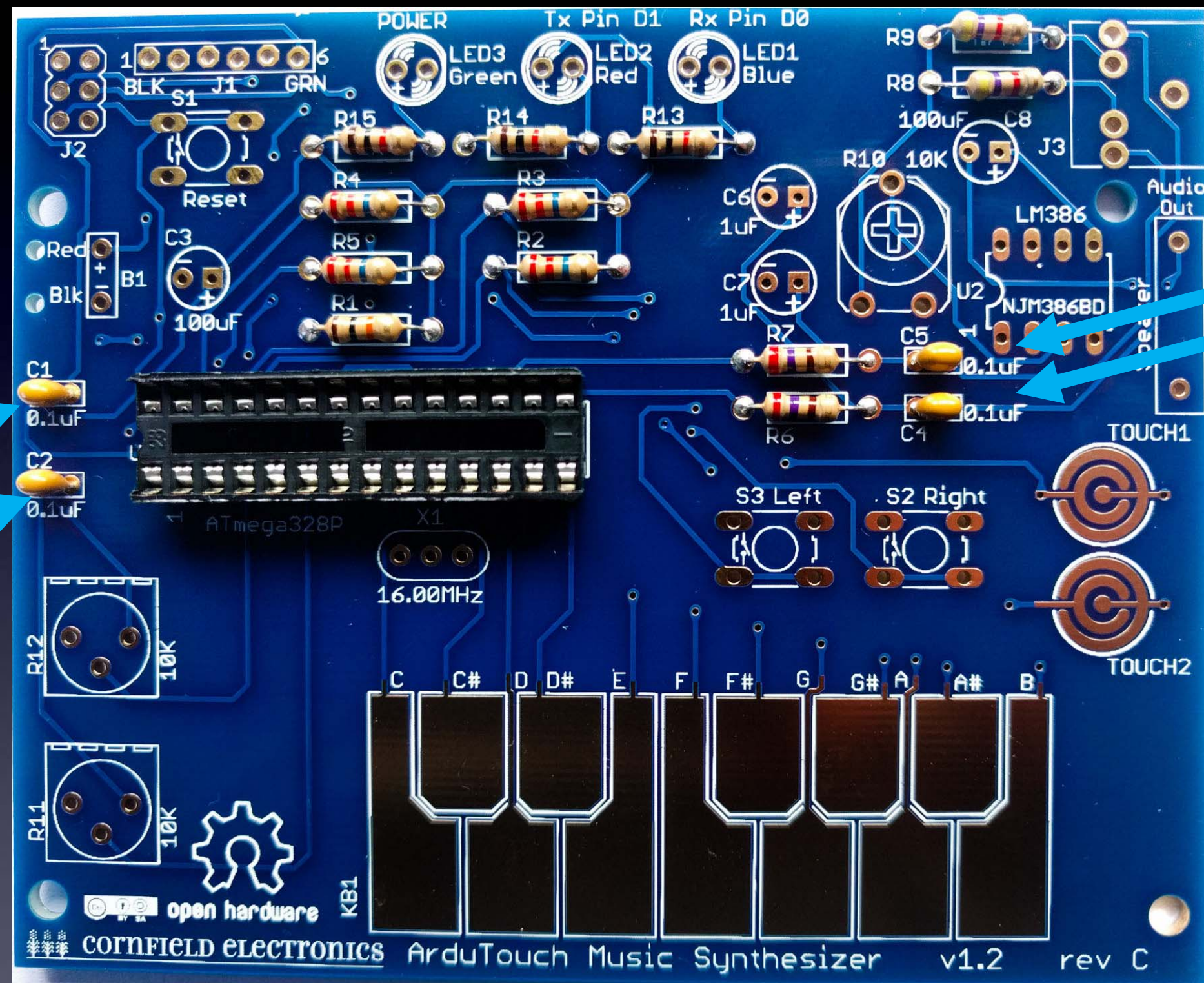
bend pins down on two corners,
and solder all 28 leads to the board

U1: microcontroller socket

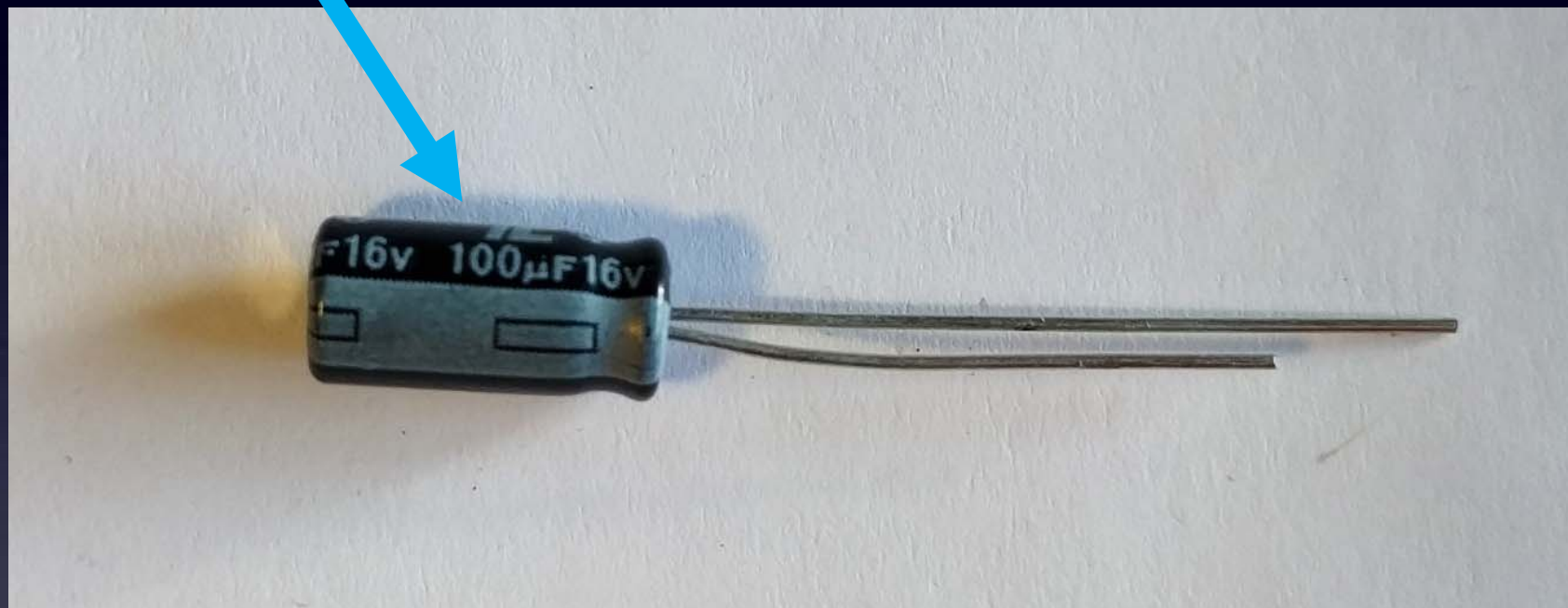


All 28 leads soldered to the board:

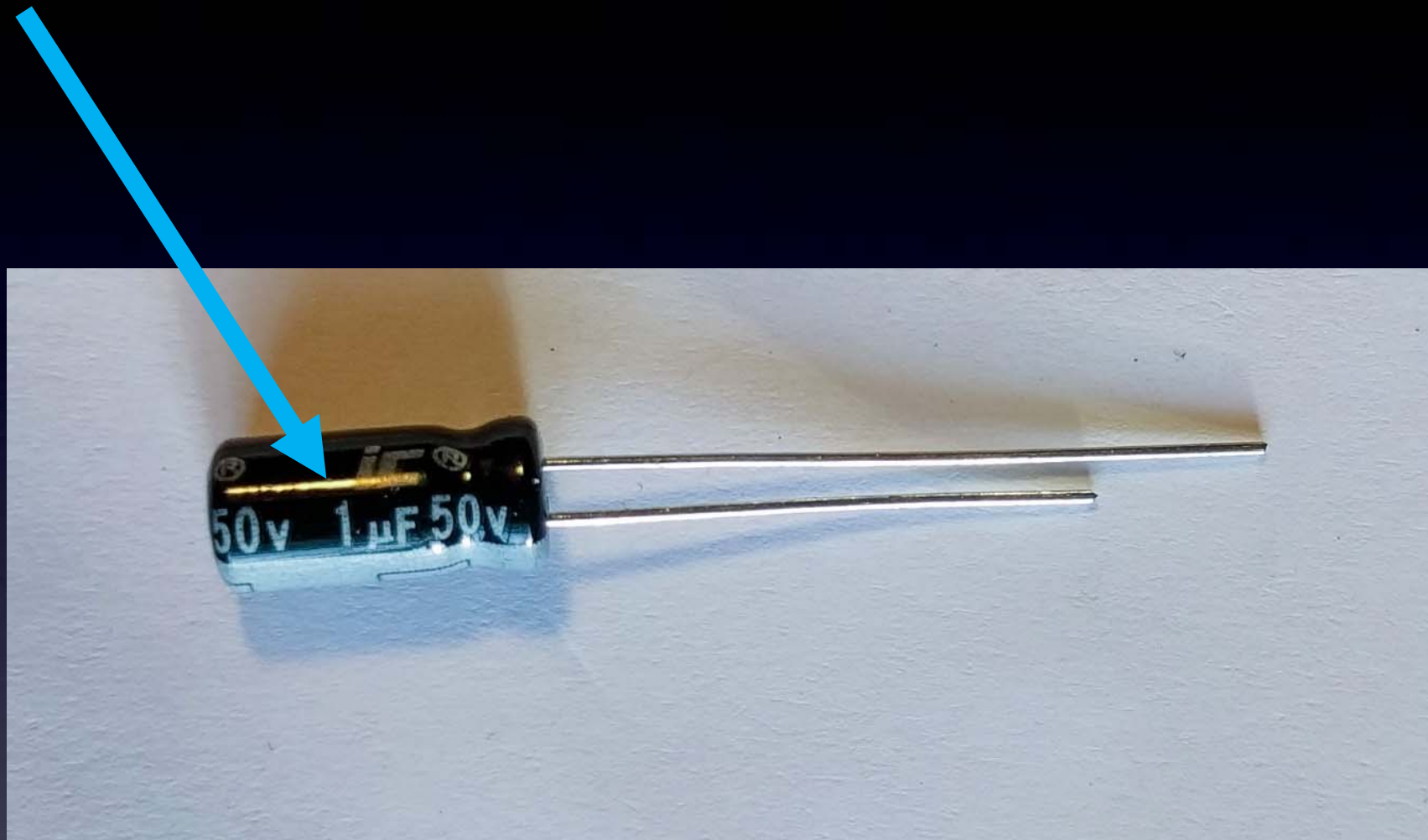
→ Notice that each pad is totally covered with solder. ←



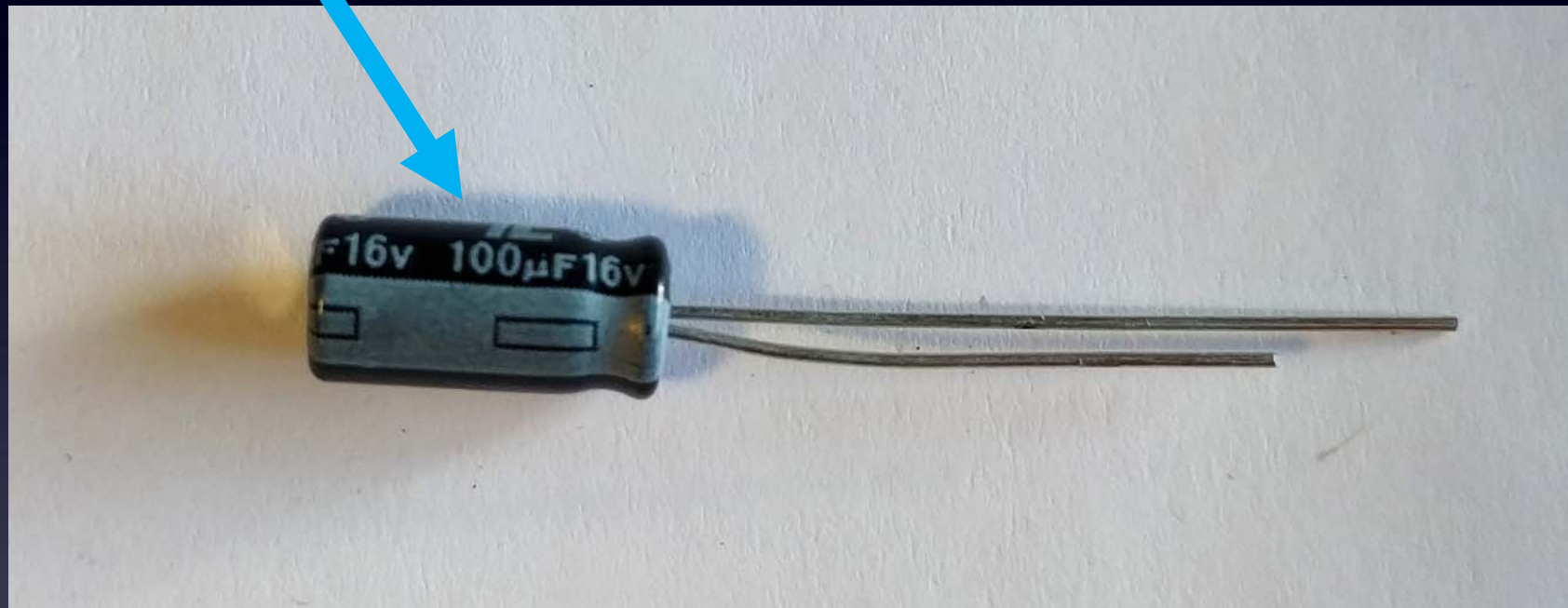
C1, C2, C4, C5



C3, C8: 100uF



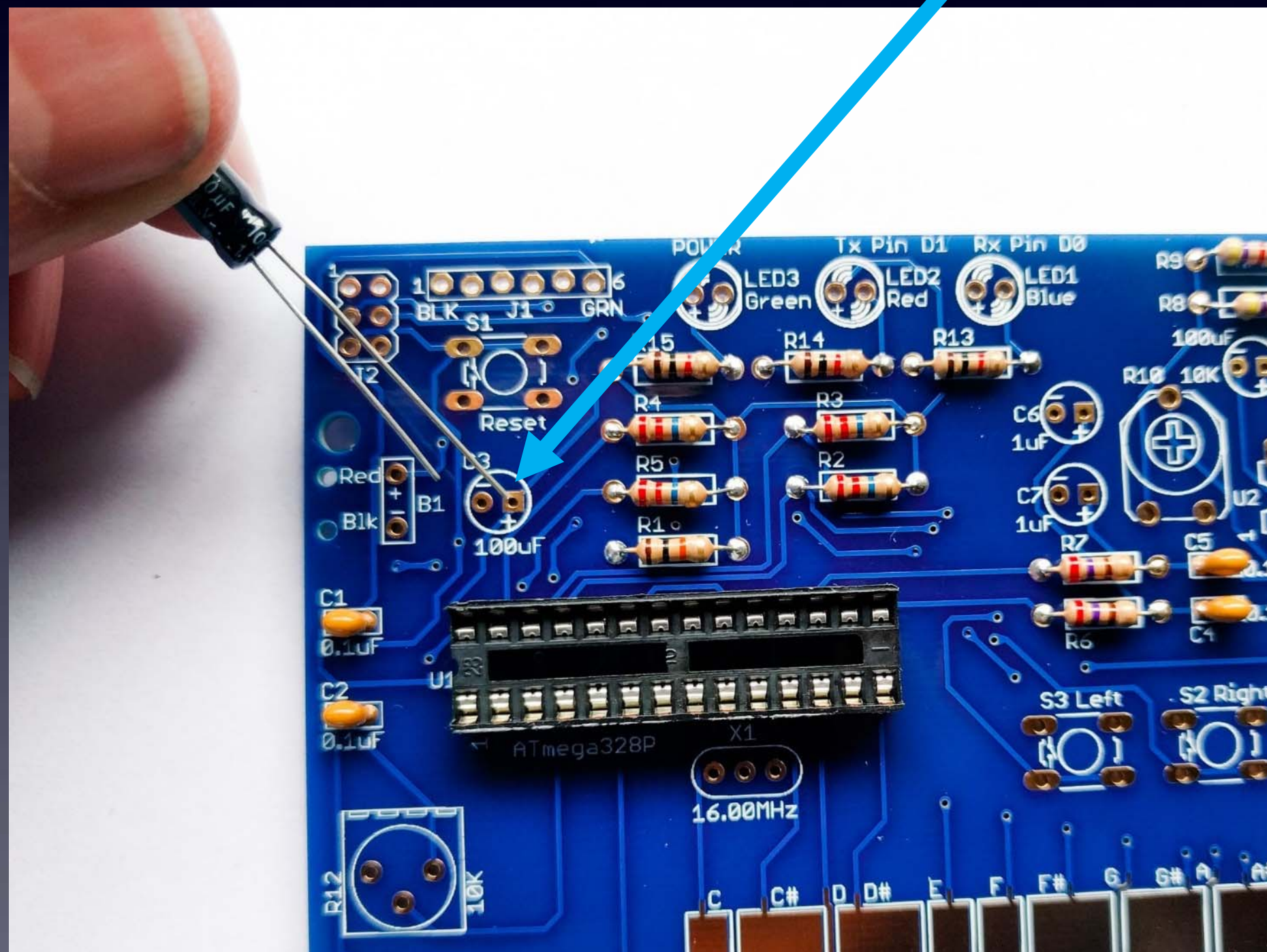
Different than C3, C8 !
C6, C7: 1uF

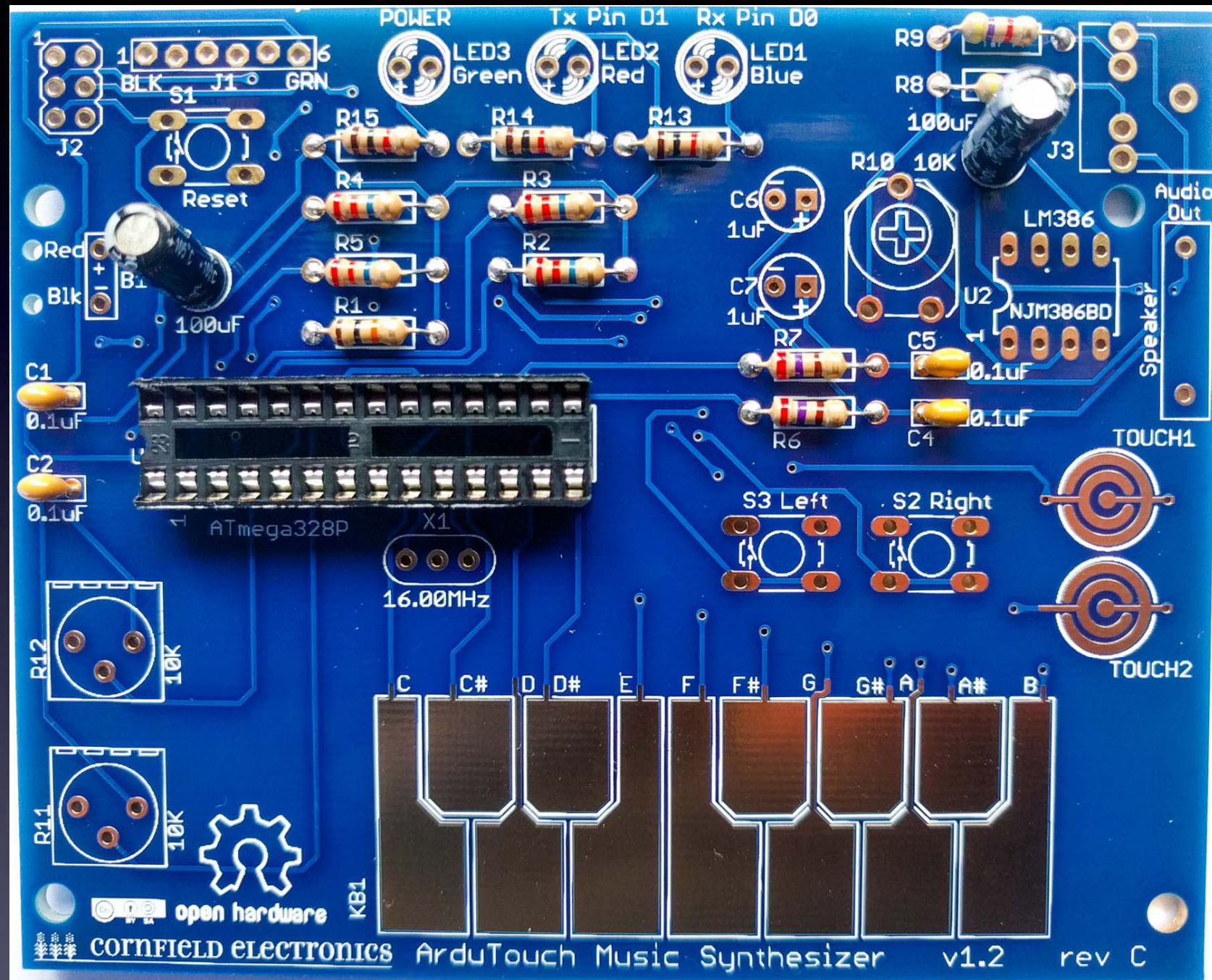


C3, C8: 100uF

C3, C8:
Long Lead “+”

Use 100uF !!

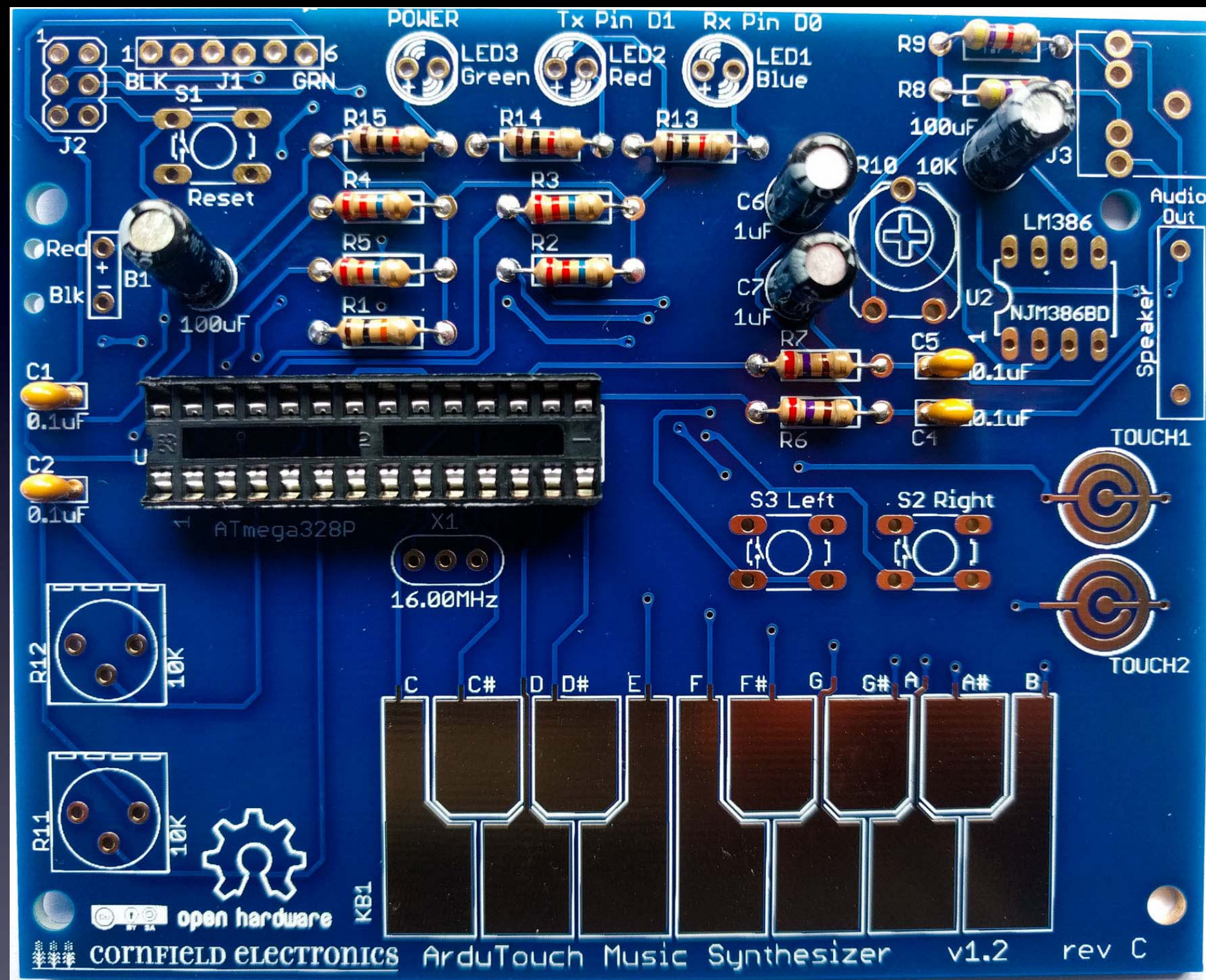




C3, C8: 100uF – soldered to board

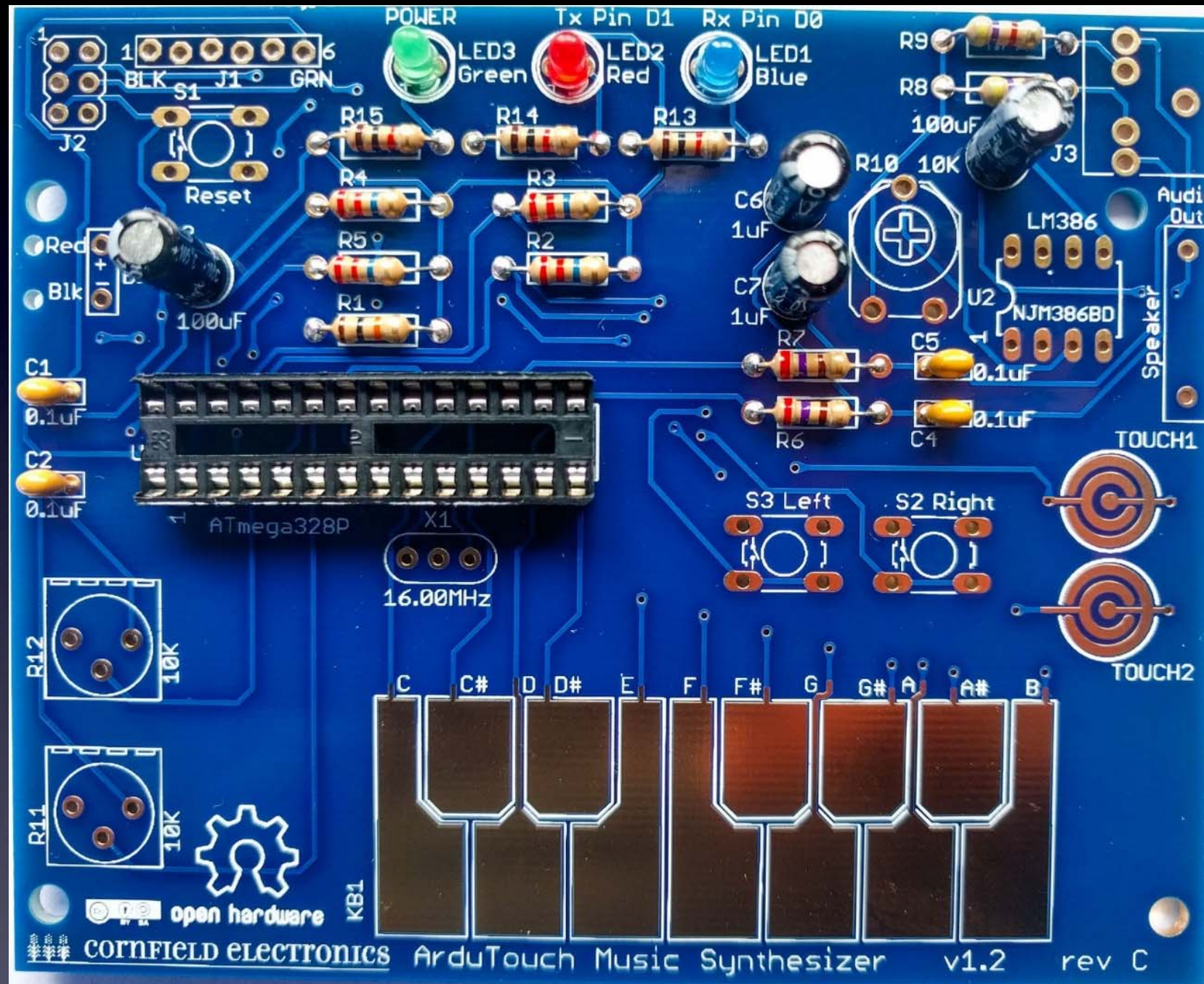


C6, C7: 1uF



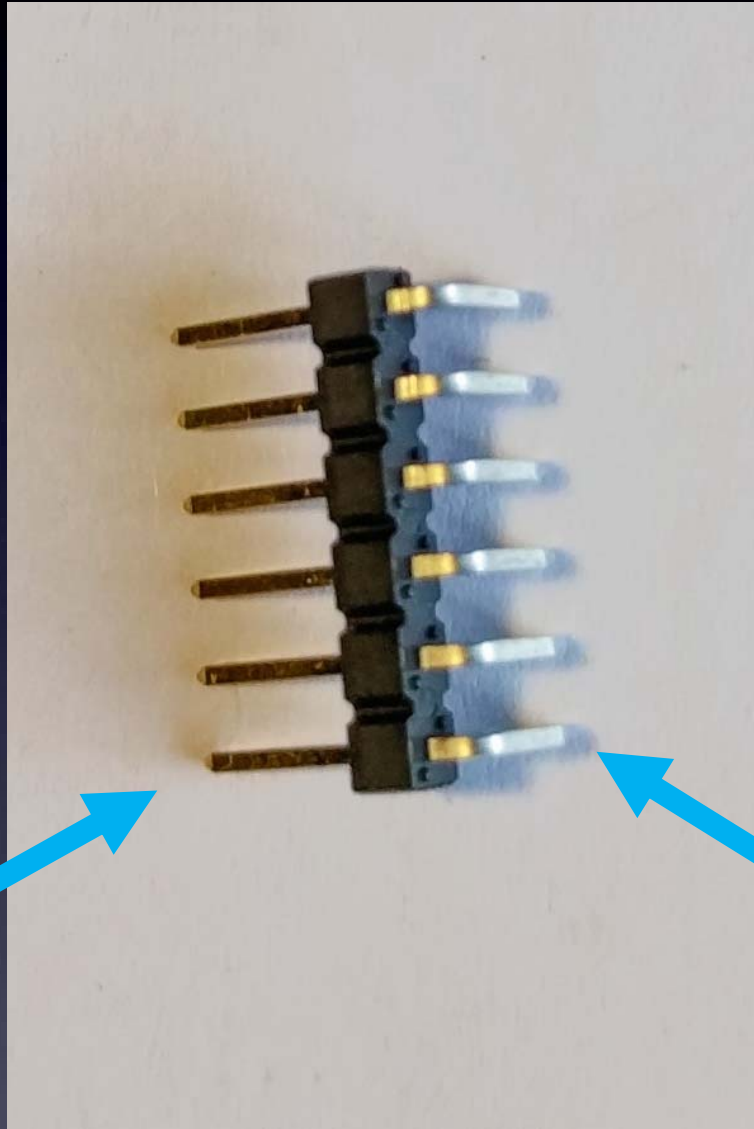
C6, C7: 1uF – soldered to board

We'll use them for the speaker



LED3, LED2, LED1

Green, Red, Blue – soldered to board



long leads

short leads

J1

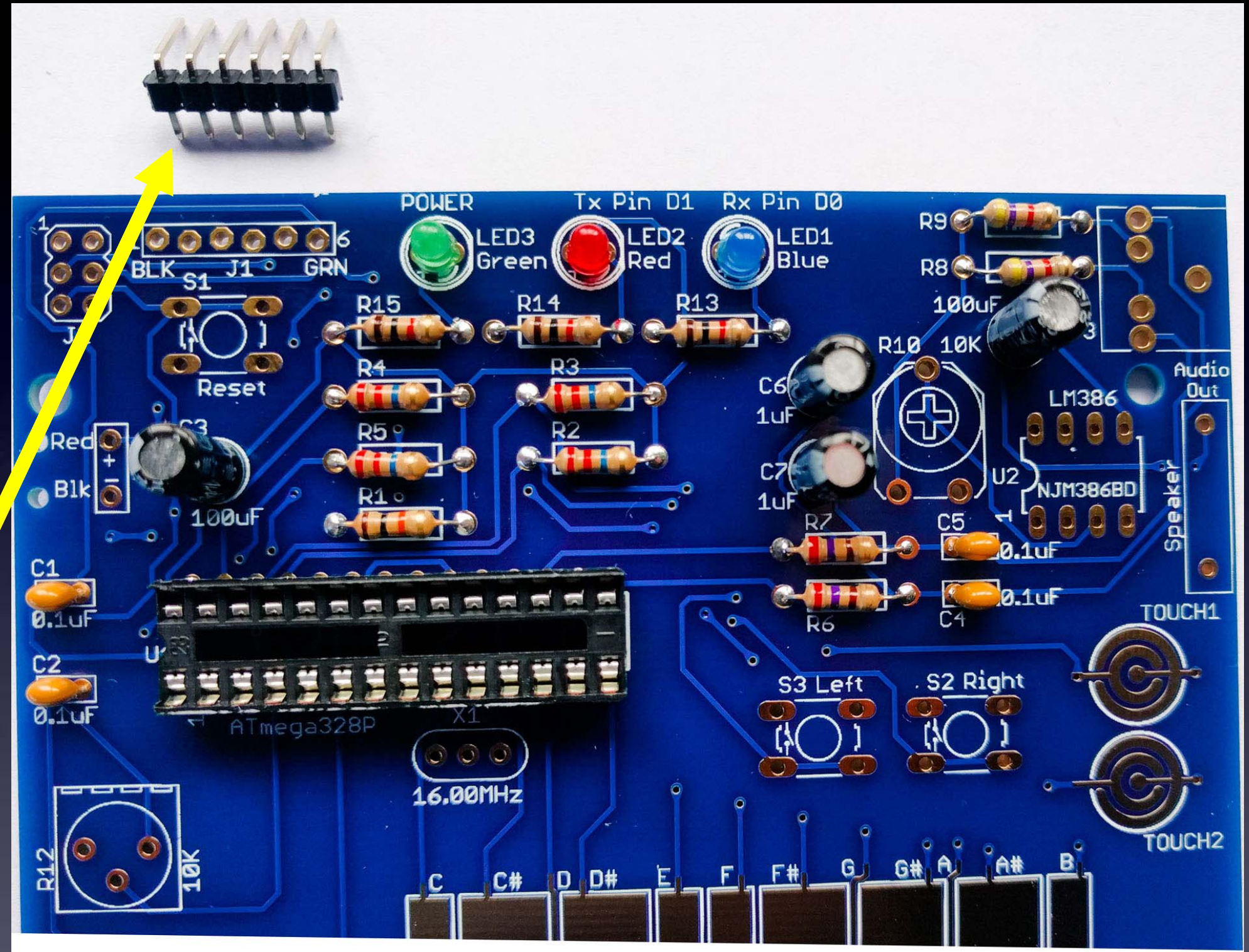
Short leads into board

J1

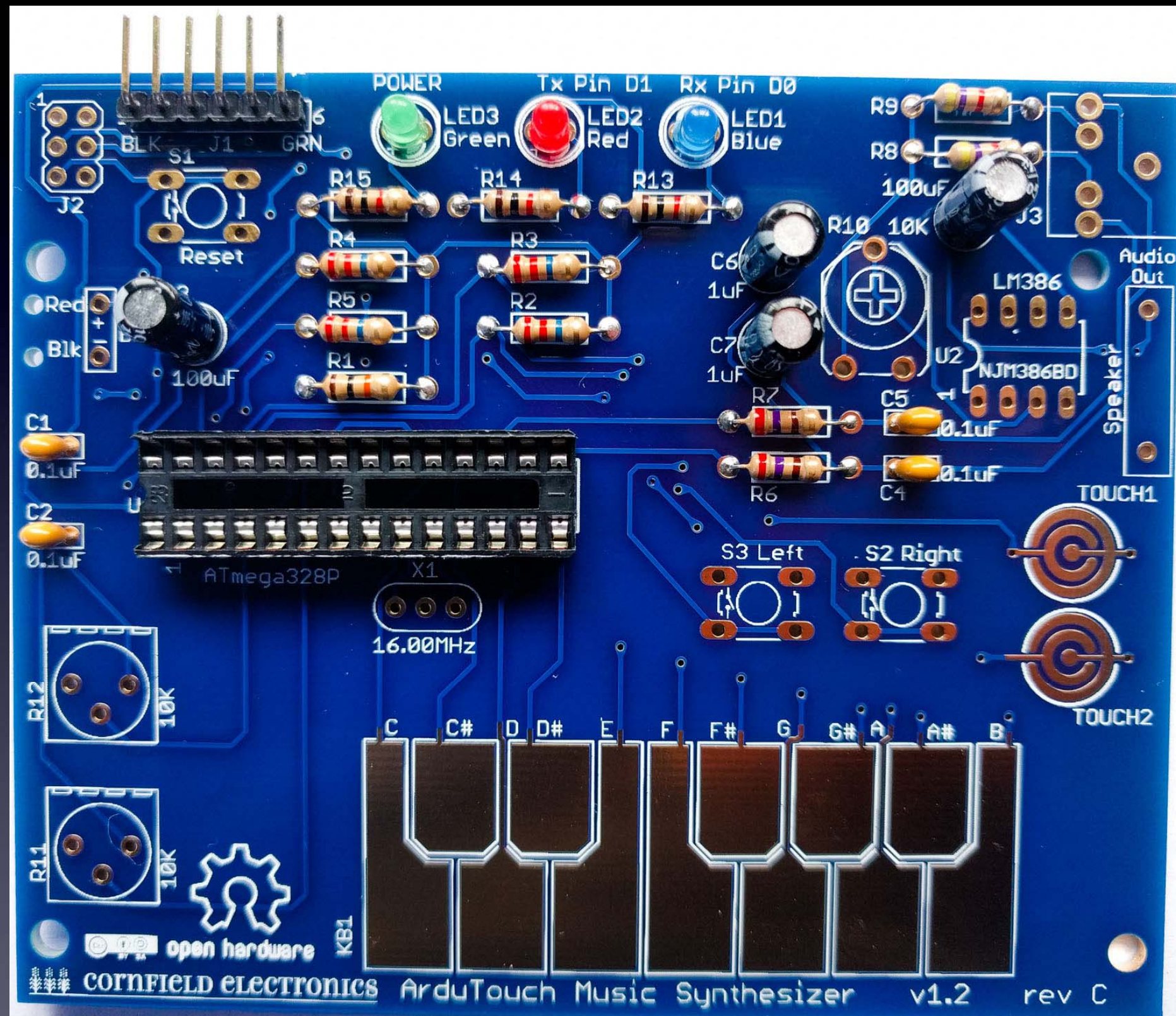
IMPORTANT:

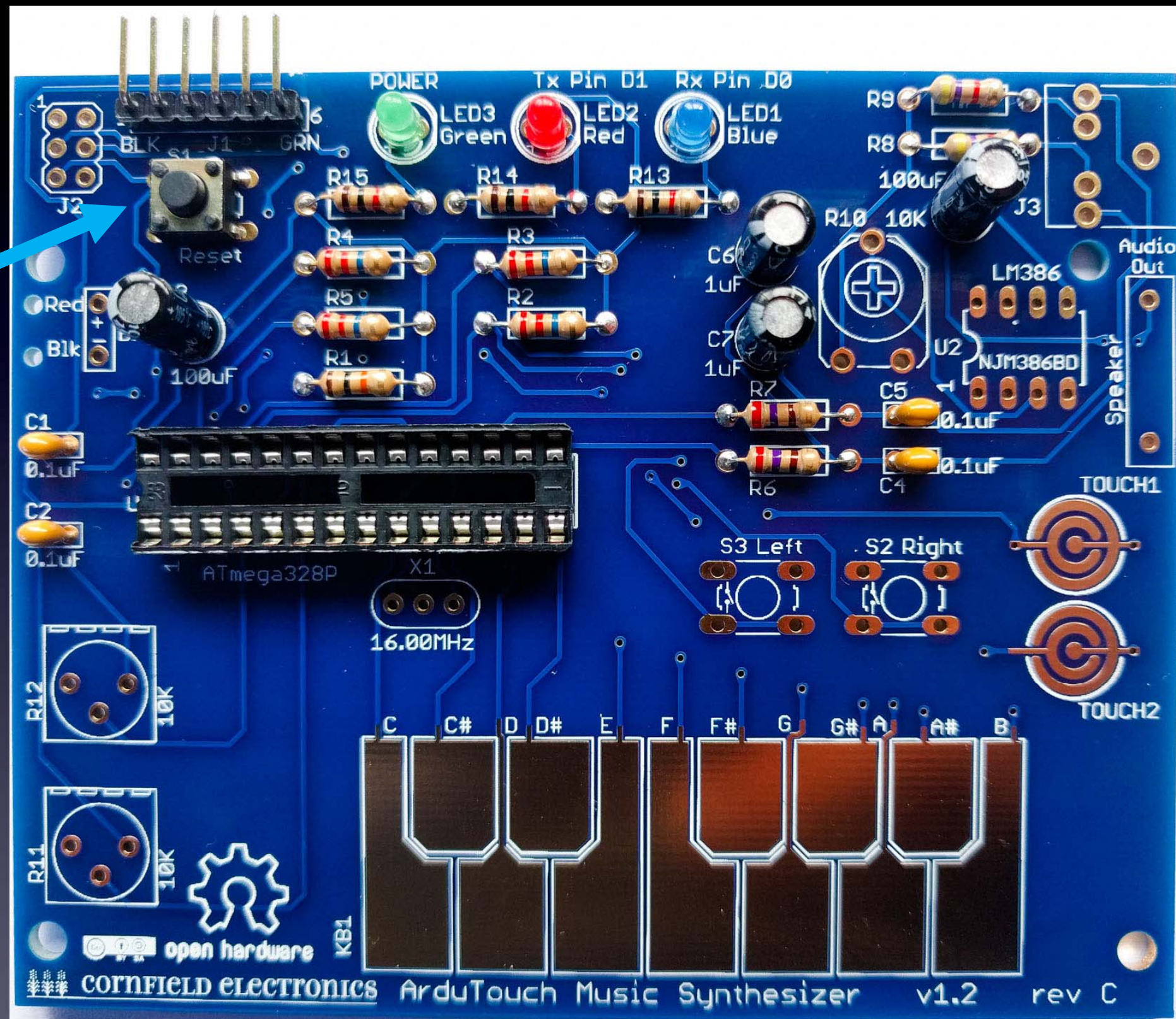
short leads
go into the board

→ long leads sticking out from
board



J1



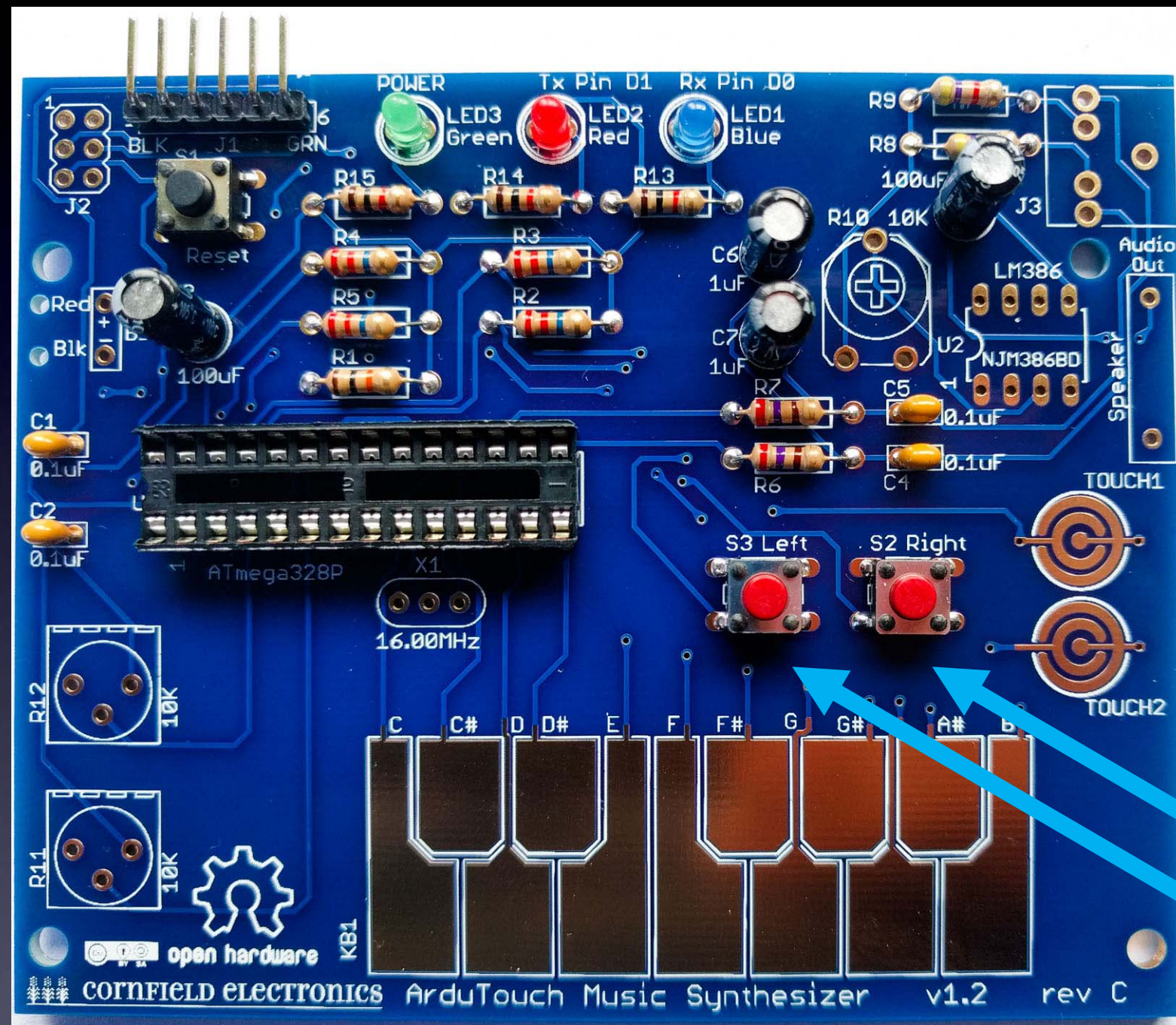


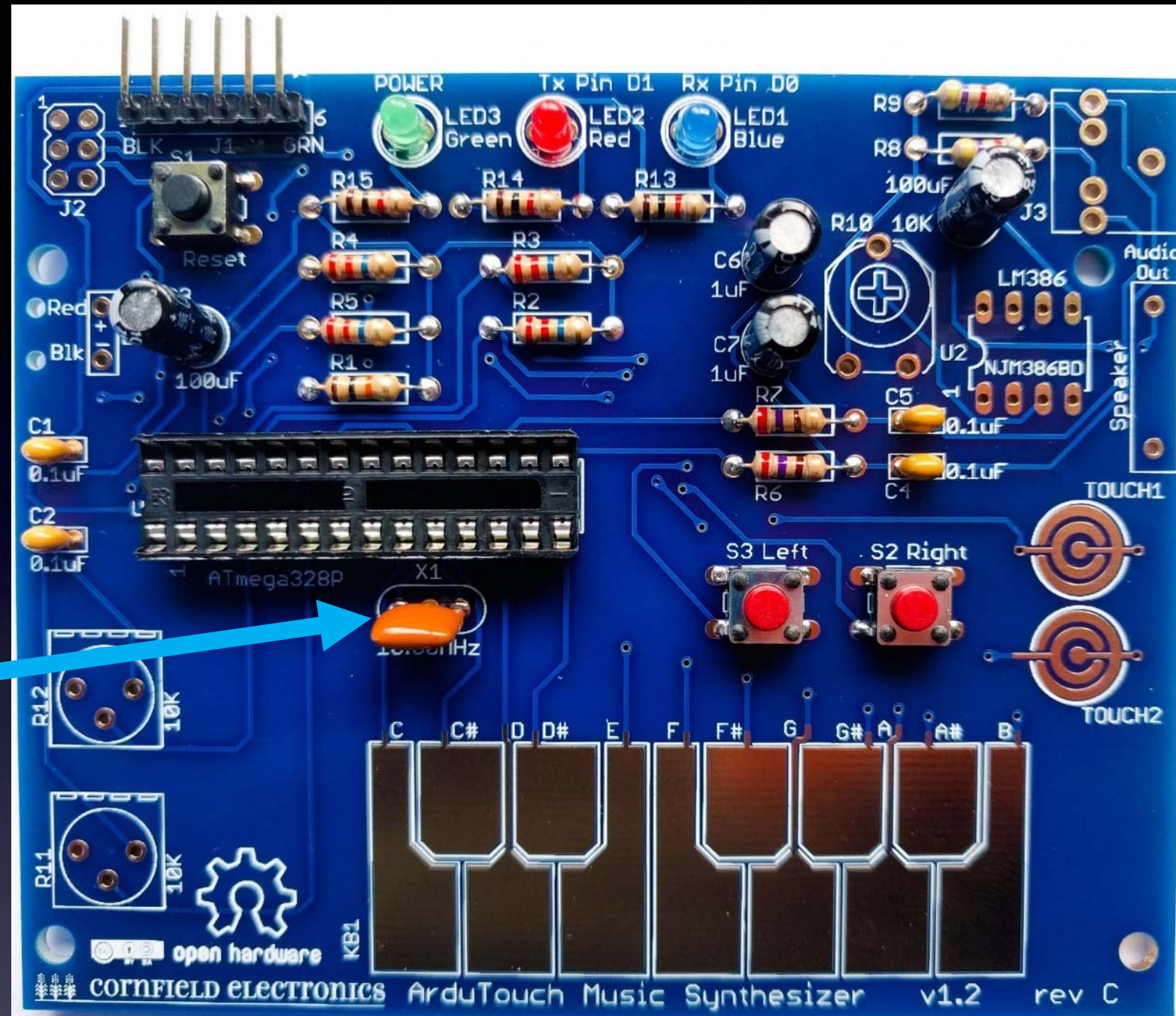
S1: black Reset button

Note: The color of this switch is not important (some kits may have different colors).

S2, S3: Red buttons

Note: The color of these switches is not important (some kits may have different colors).



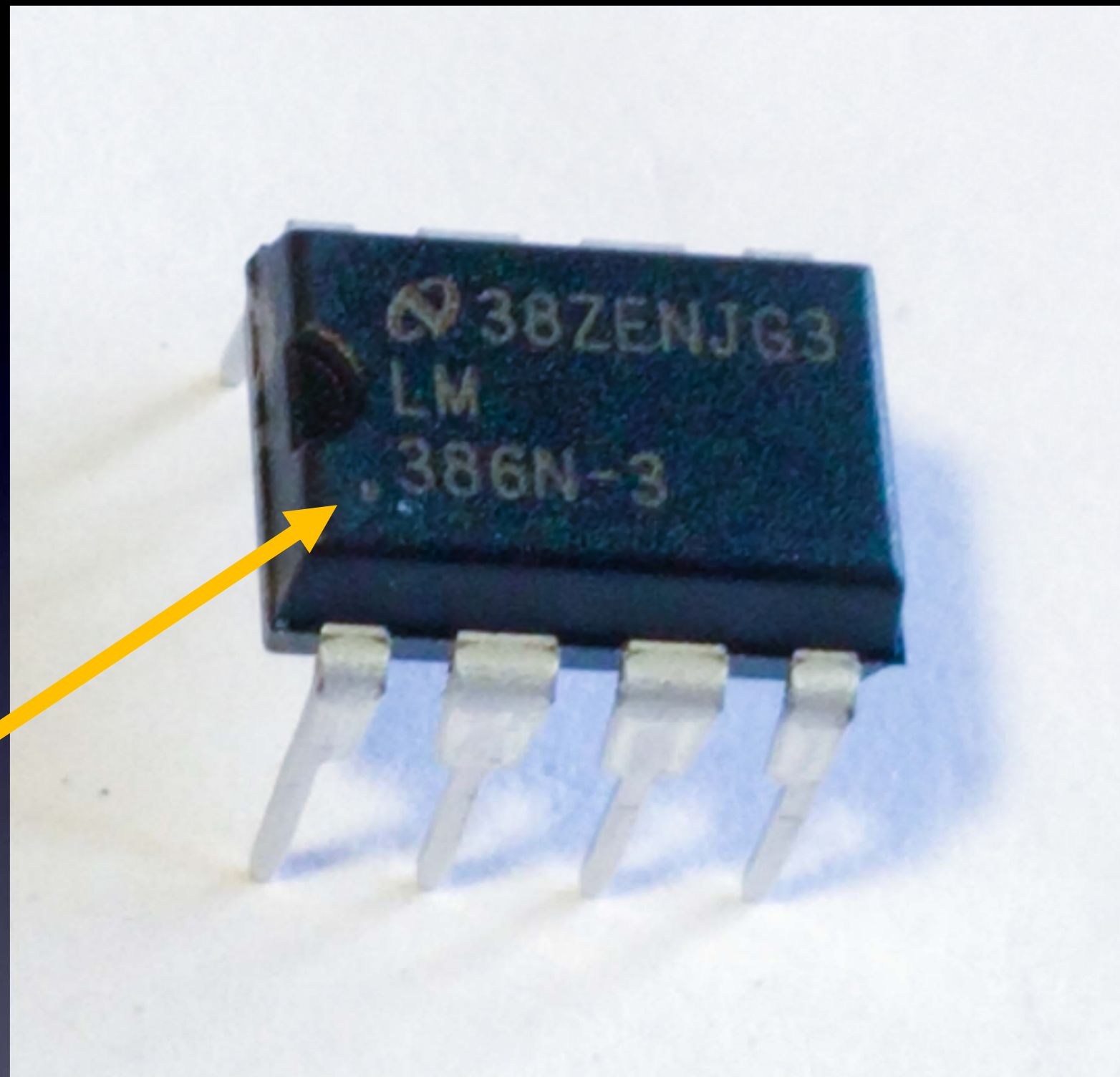


X1

The orientation of X1 does not matter.

Note: X1 may be yellow or blue.

U2

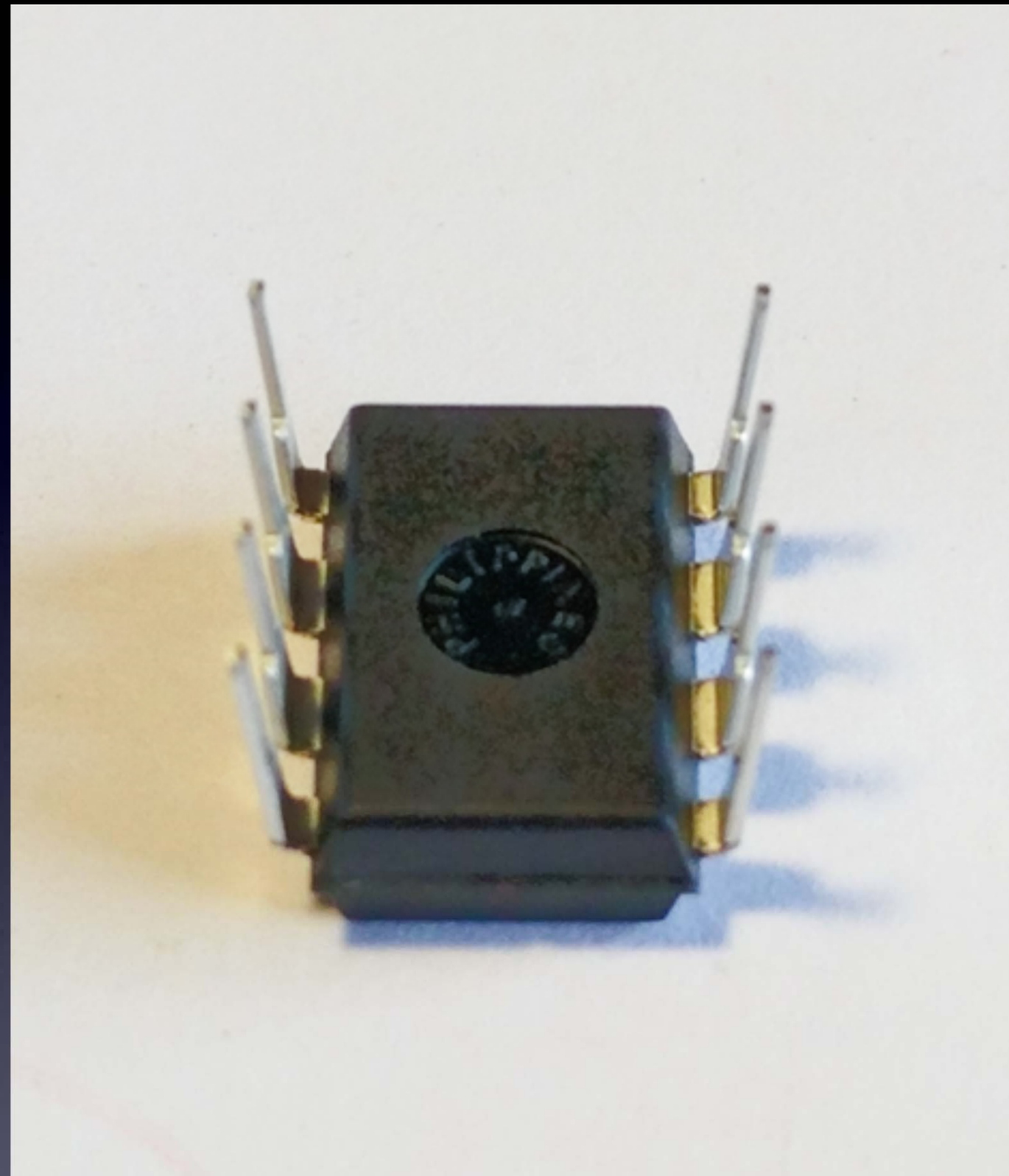


Indented black dot
Pin 1

Note: Your chip may be marked differently, but “386” will be printed on it somewhere.

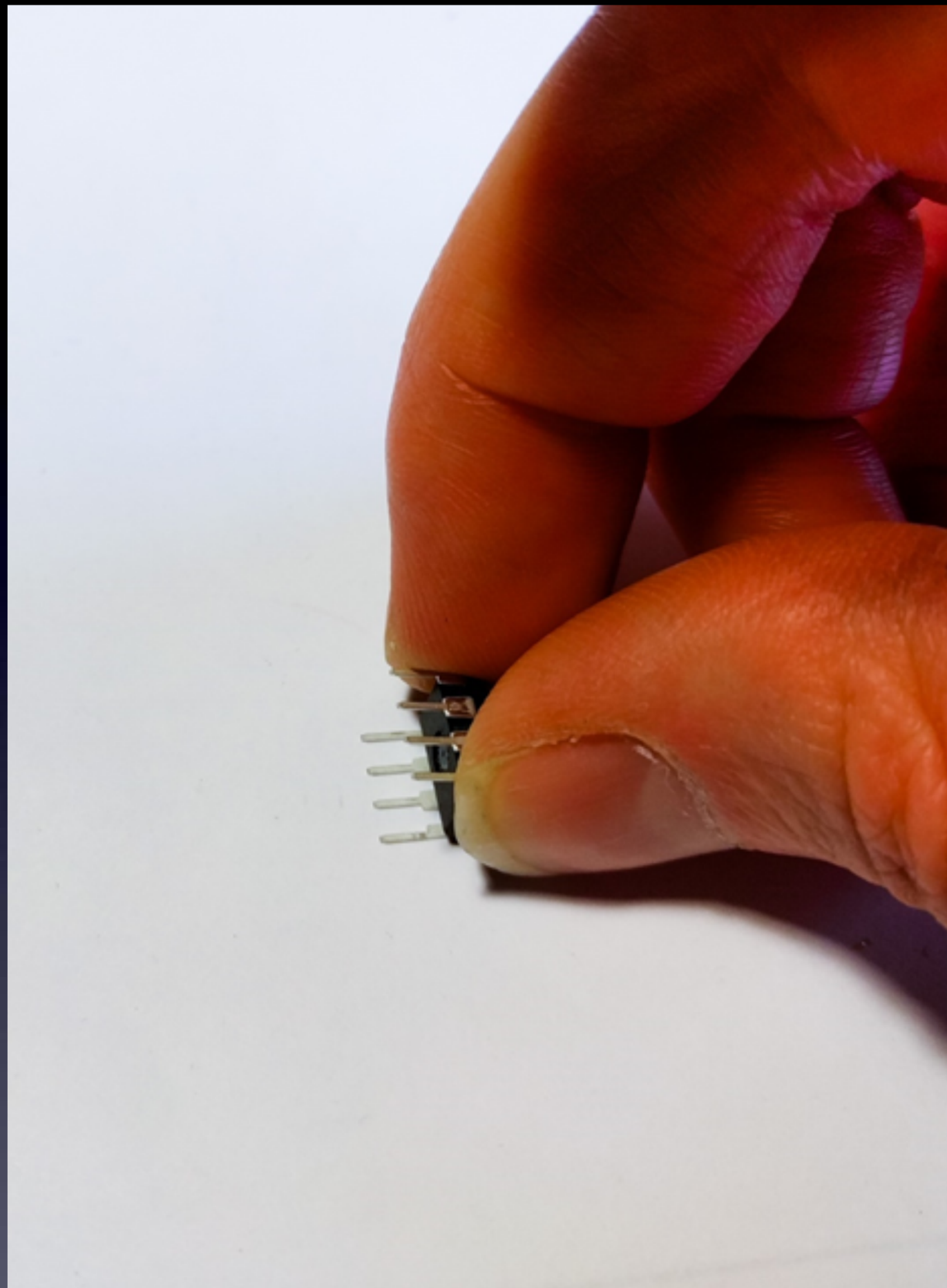
Note: Your chip may or may not have the indented half-moon at the left,
it may have a black indented dot at the lower-left corner showing Pin 1.

U2



**When chips are new,
their pins are bent out.**

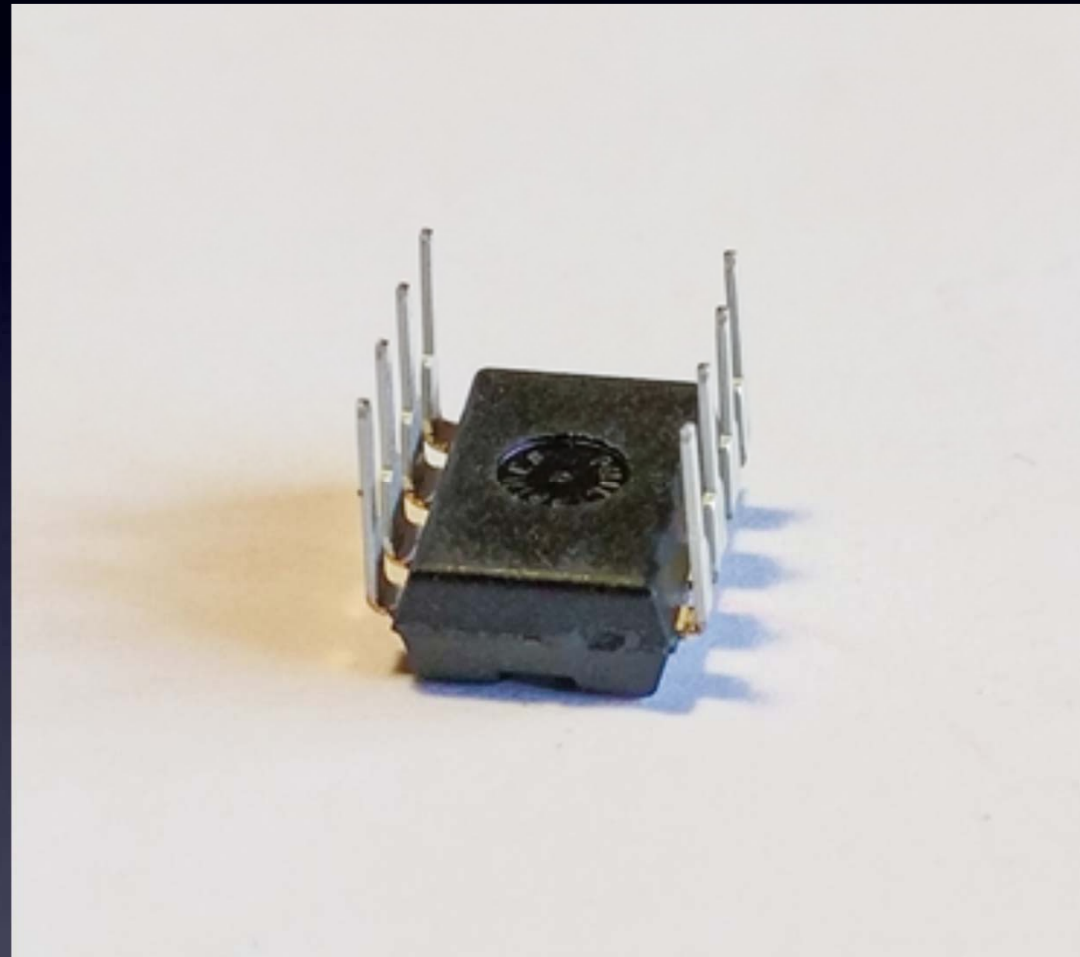
U2

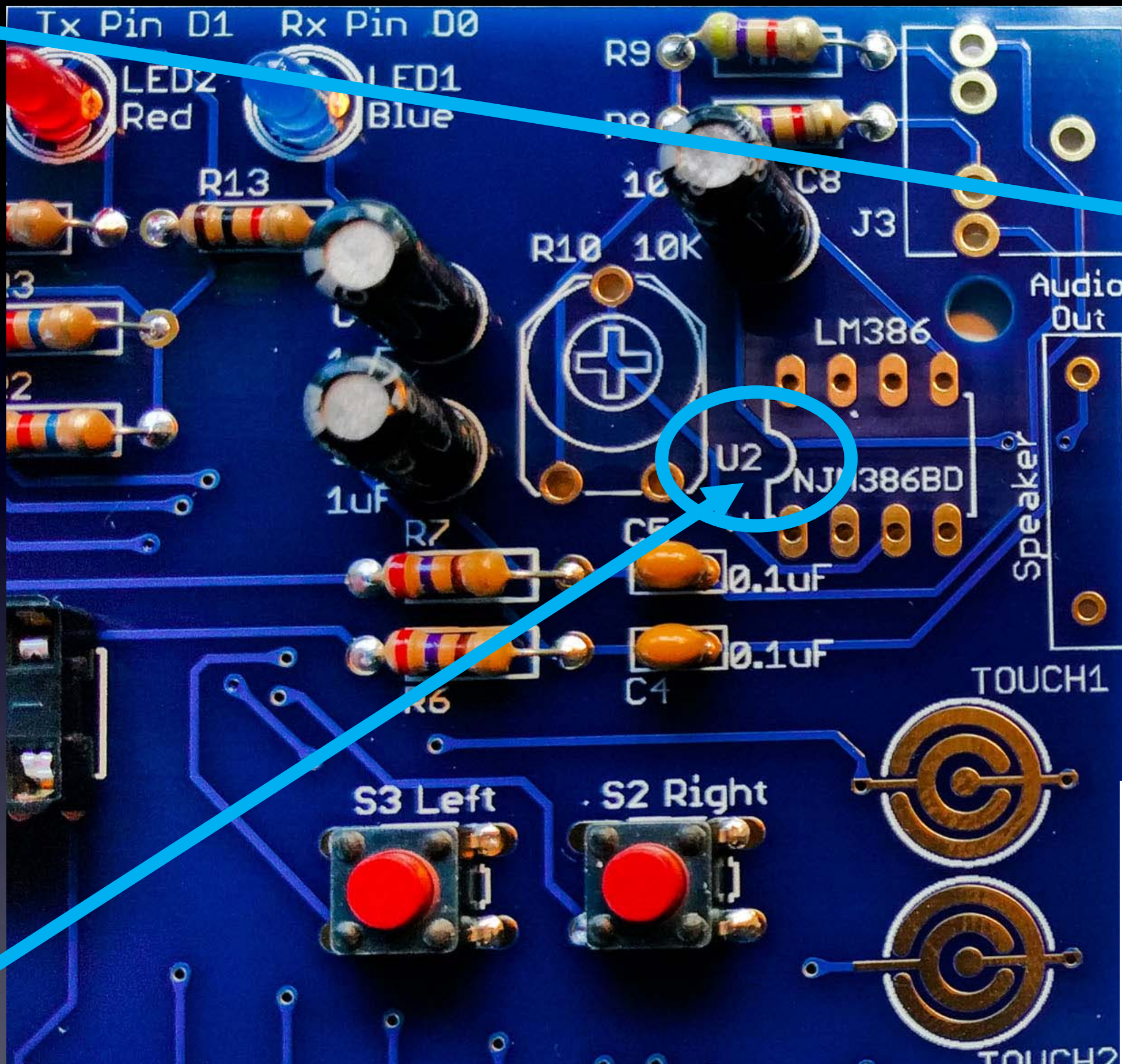


**We need the pins bent straight and parallel.
Use your work table to (gently) bend the leads.**

U2

**Gently
bend leads
so they're straight
and parallel**



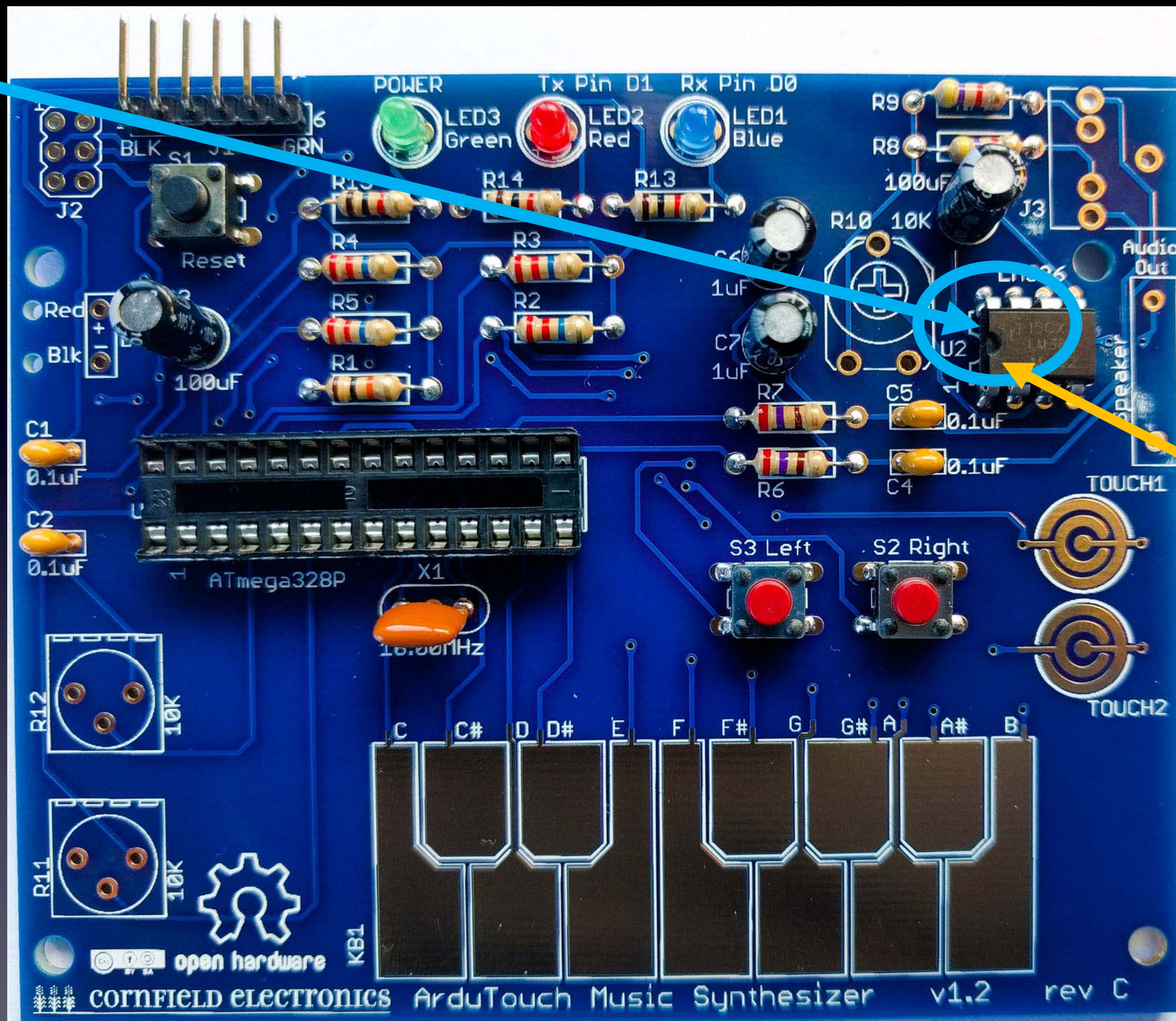


Indented black dot
Pin 1

**proper
orientation**

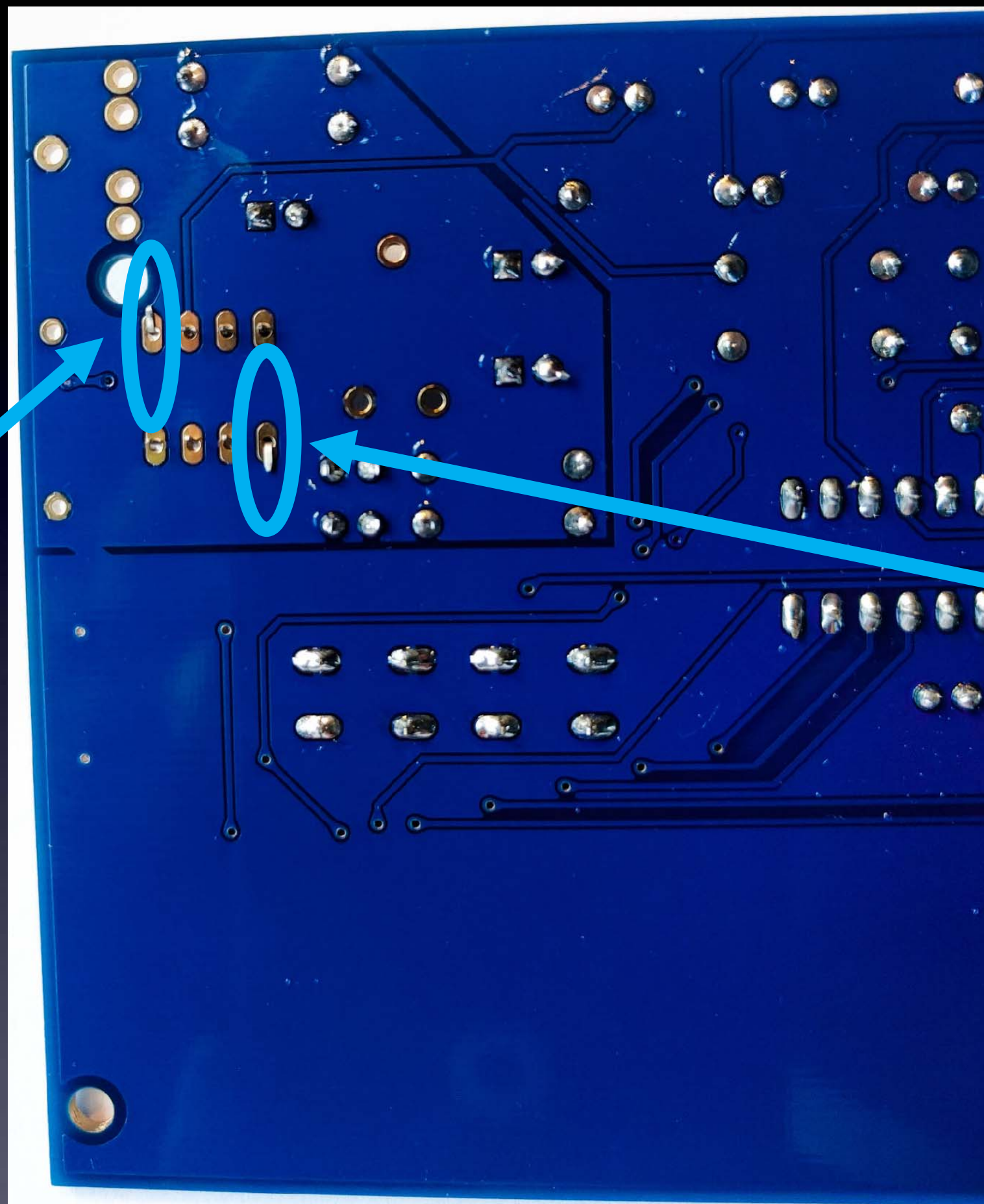
Note: Your chip may or may not have the indented half-moon at the left, it may have a black indented dot at the lower-left corner showing Pin 1.

U2: audio amp chip

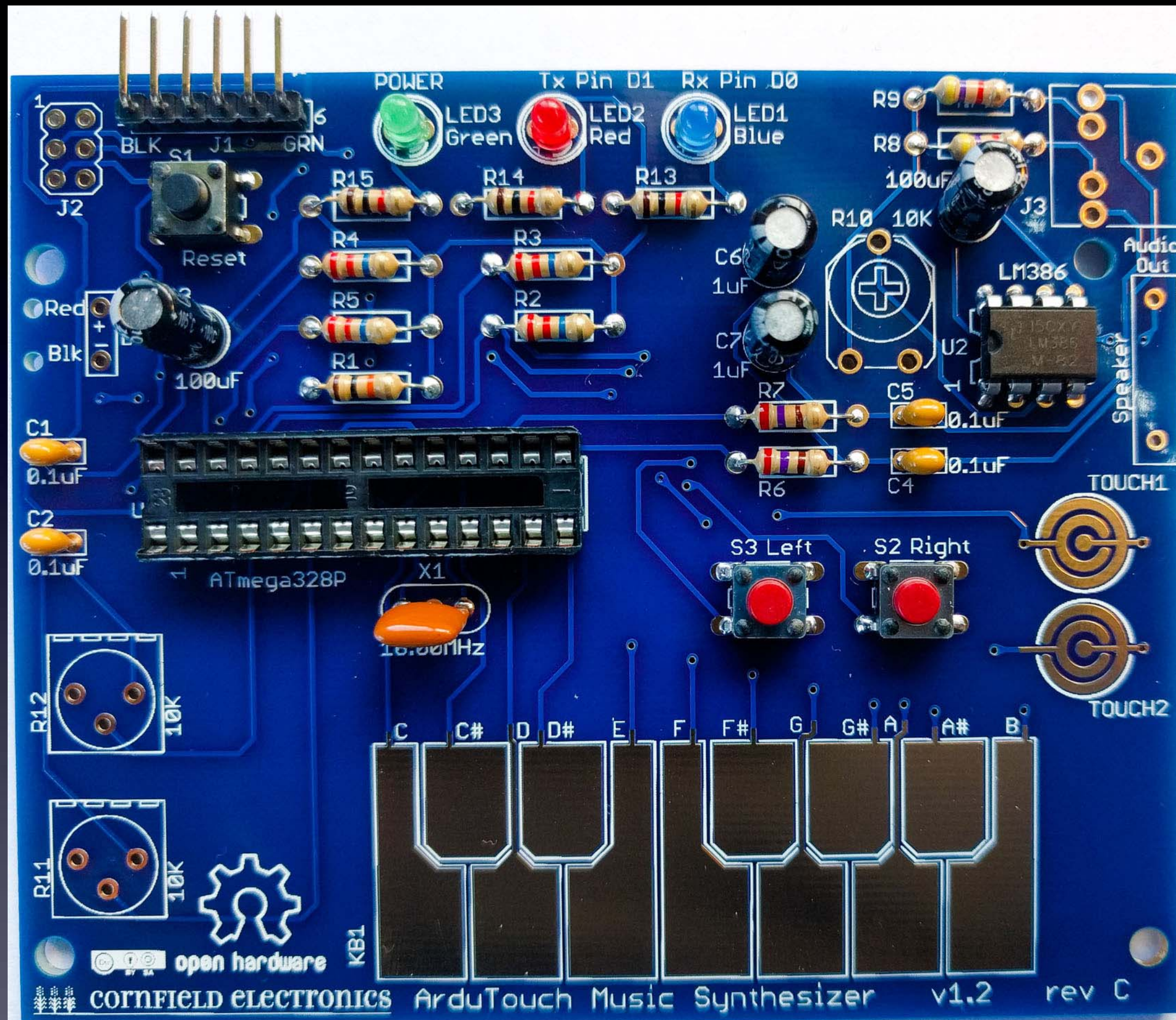


U2: inserted correctly

U2

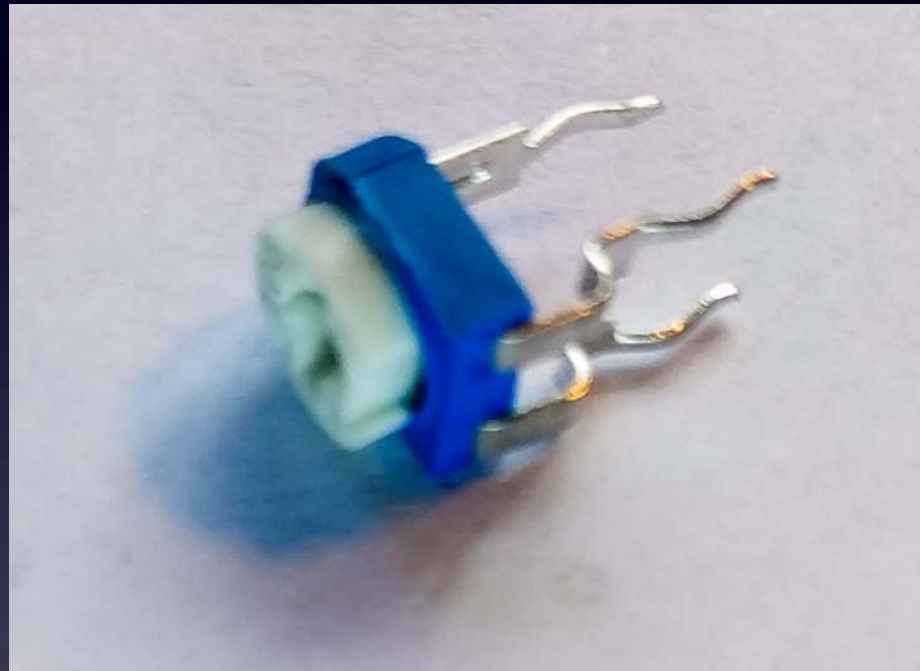


bend pins down on two corners,
and solder all 8 leads to the board



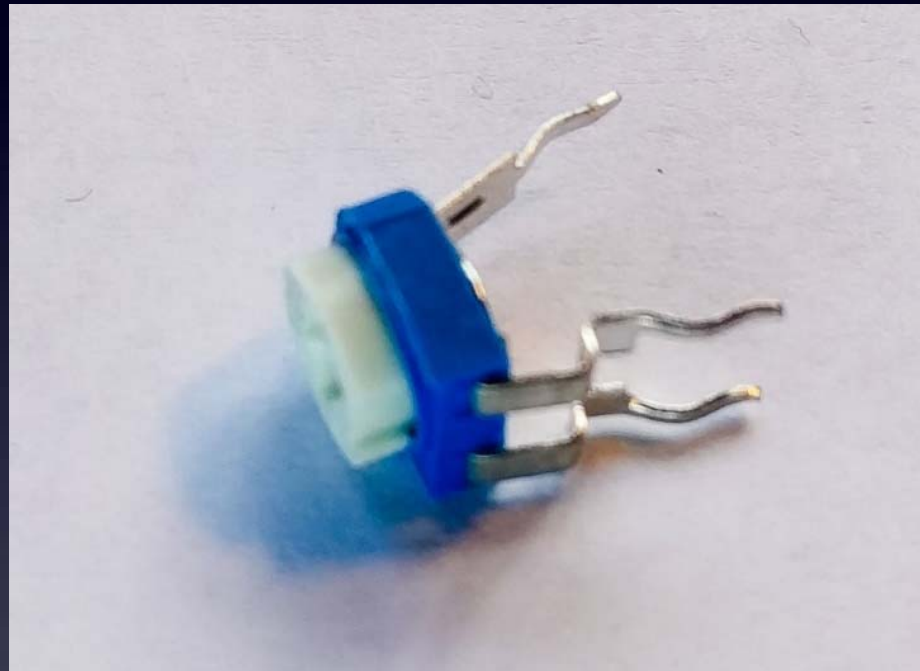
U2 – soldered to board

R10: volume control



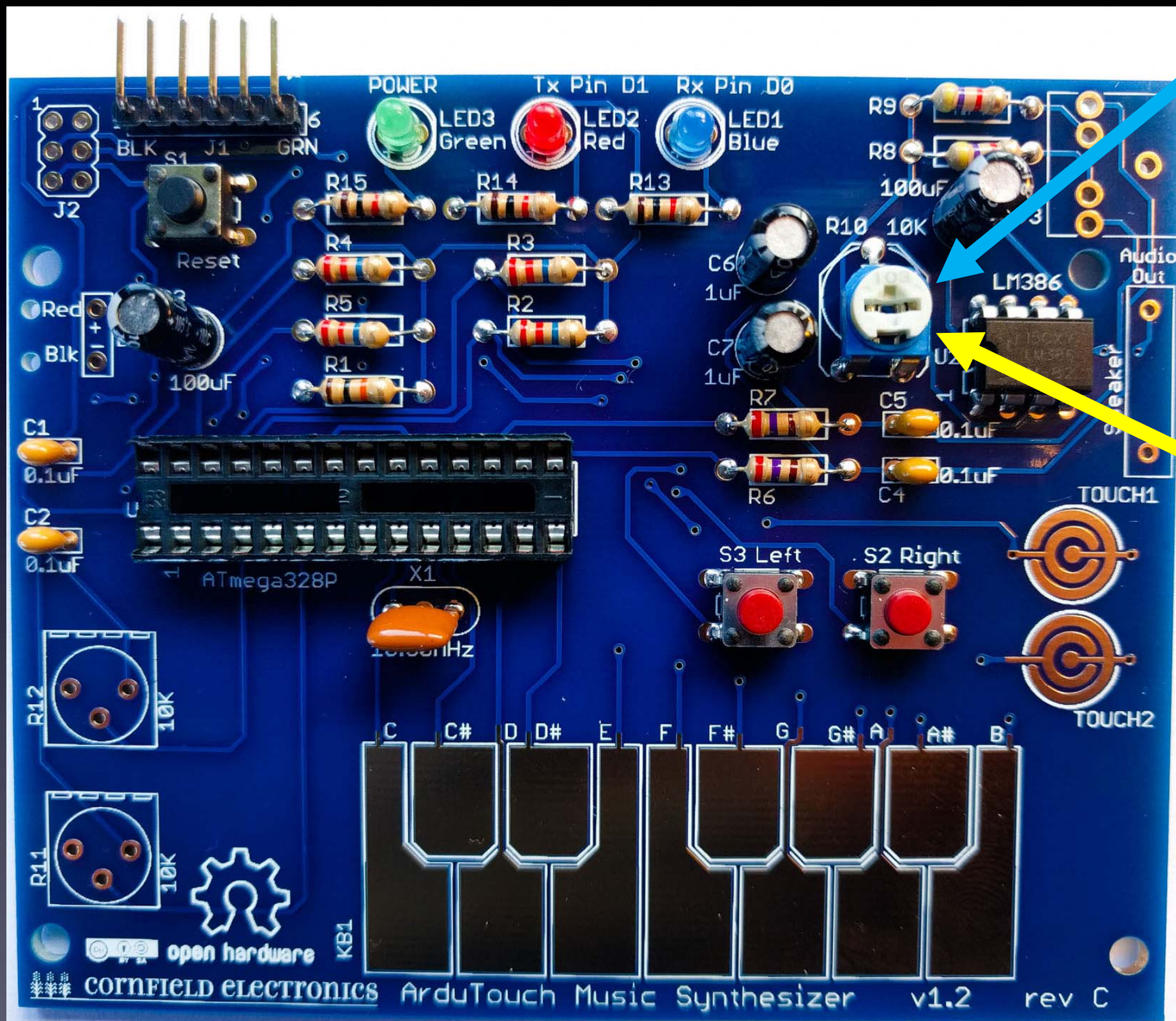
When new, the pins point straight down.

R10: volume control

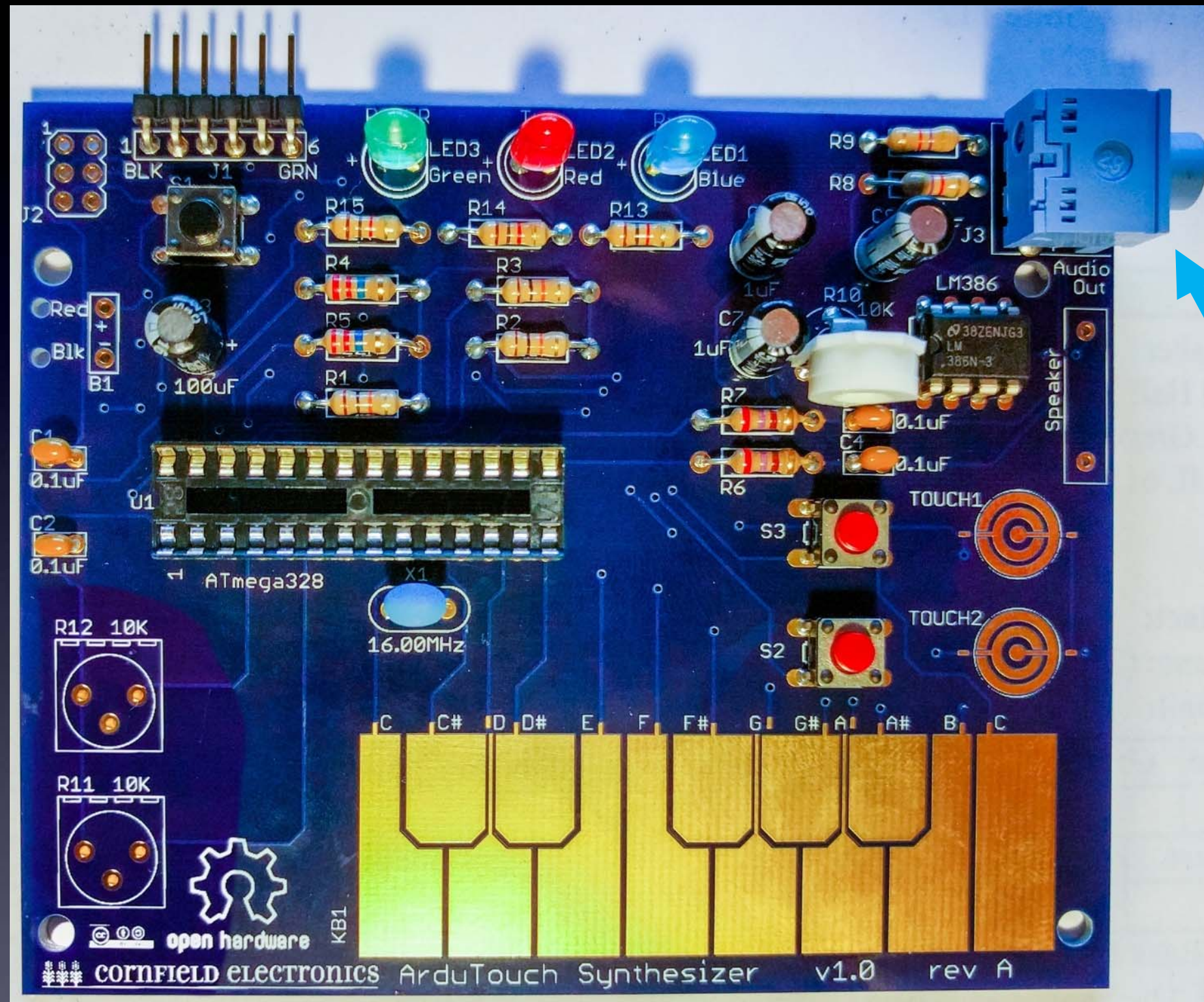


We need to bend them out a little to fit into the board.

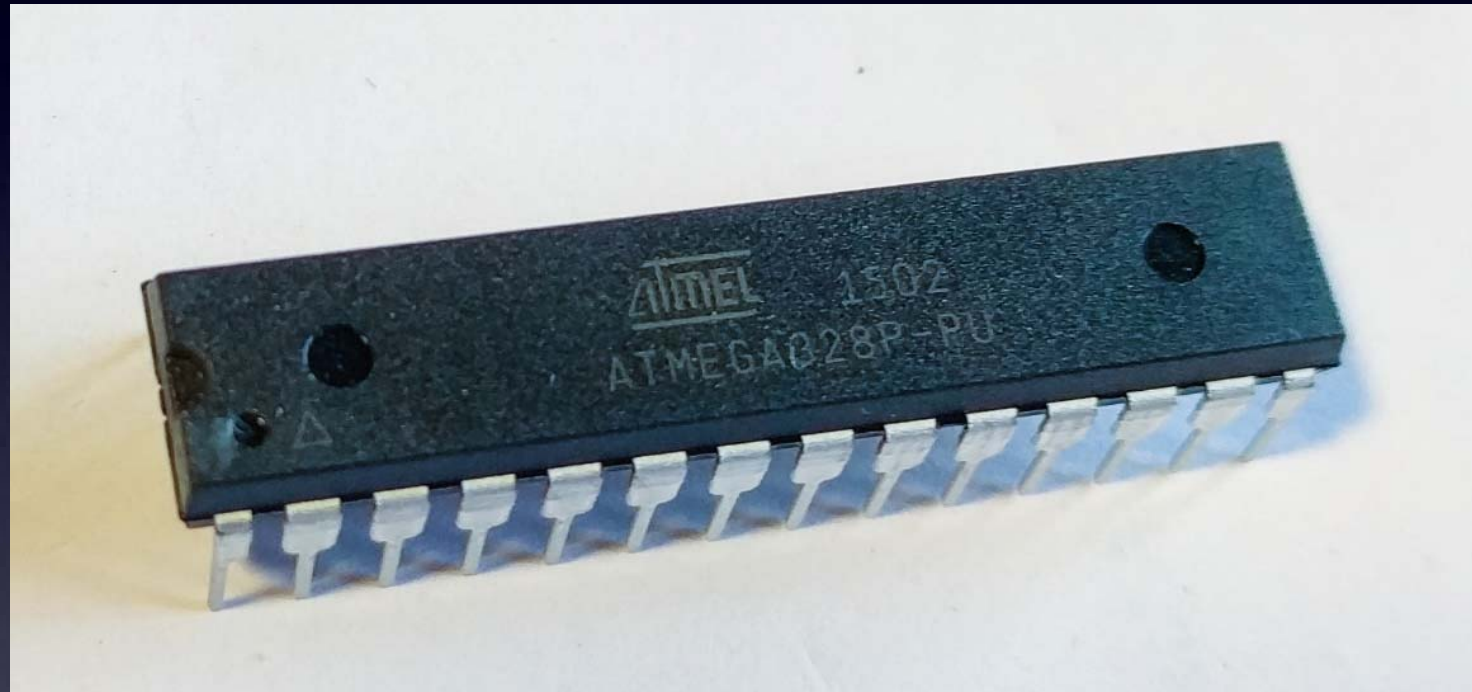
R10: volume control



If necessary,
rotate the white top
so that it looks
like this photo
(rotated half-way)

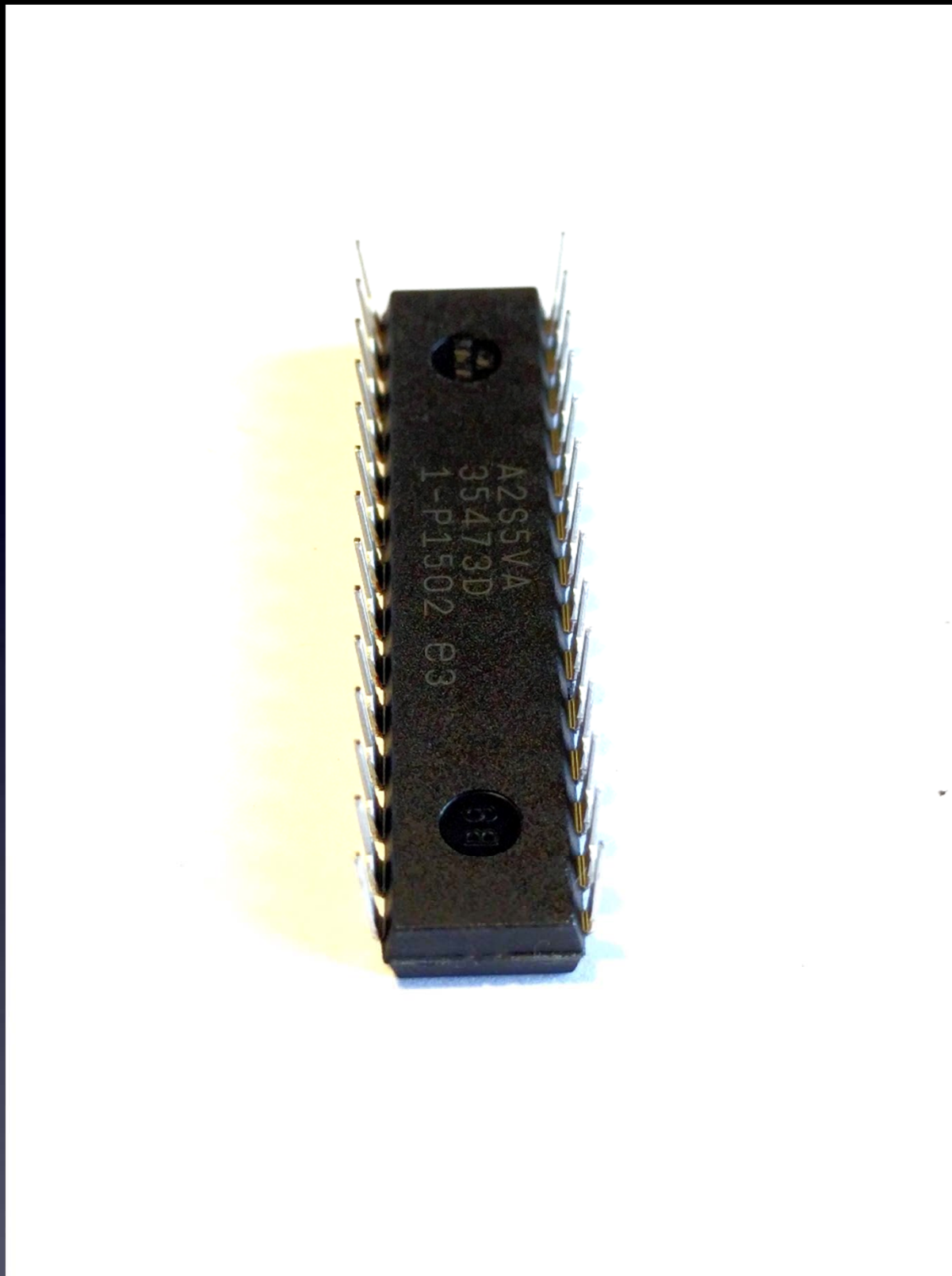


J3: headphone / output jack



U1: microcontroller

U1

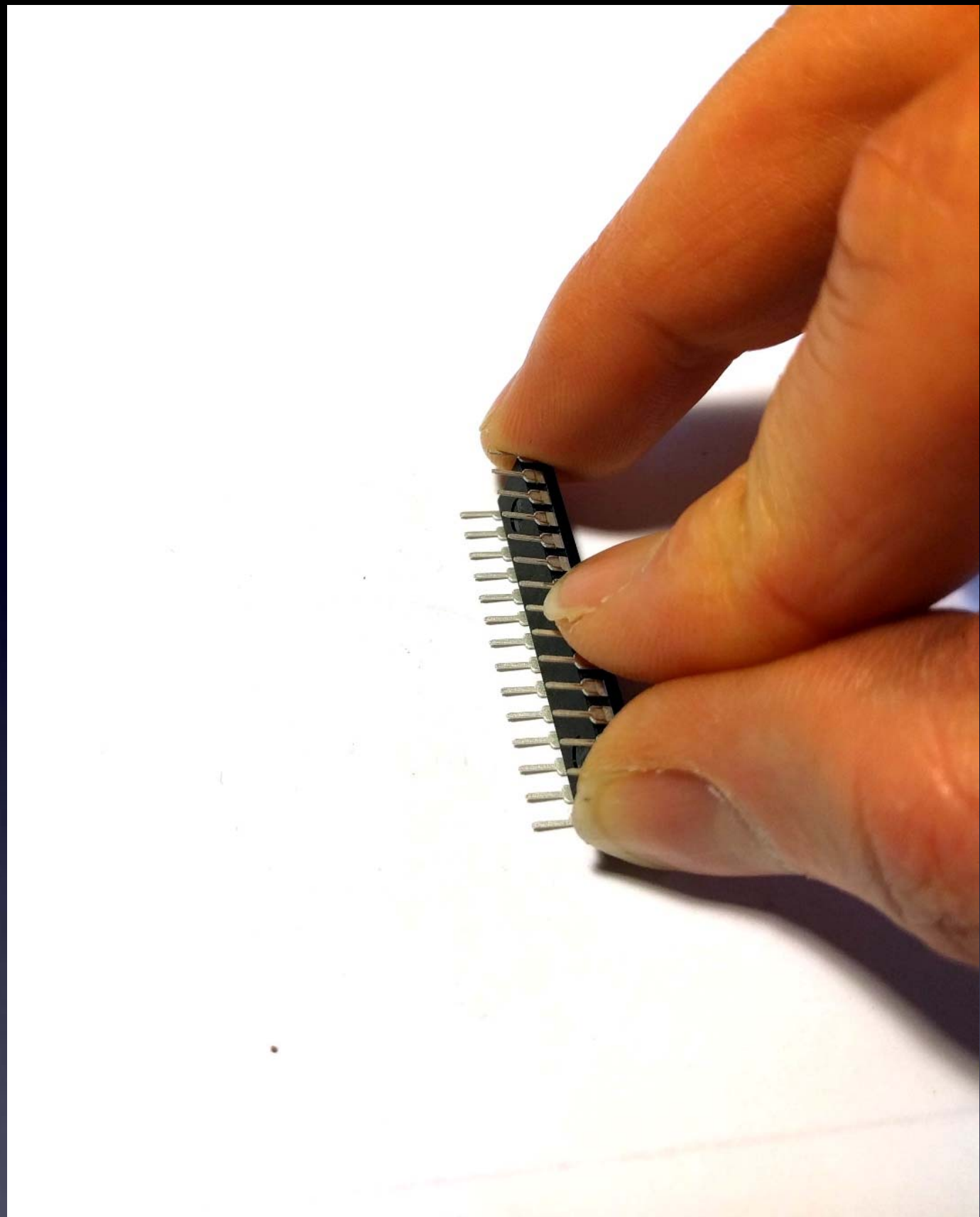


**When chips are new,
their pins are bent out.**

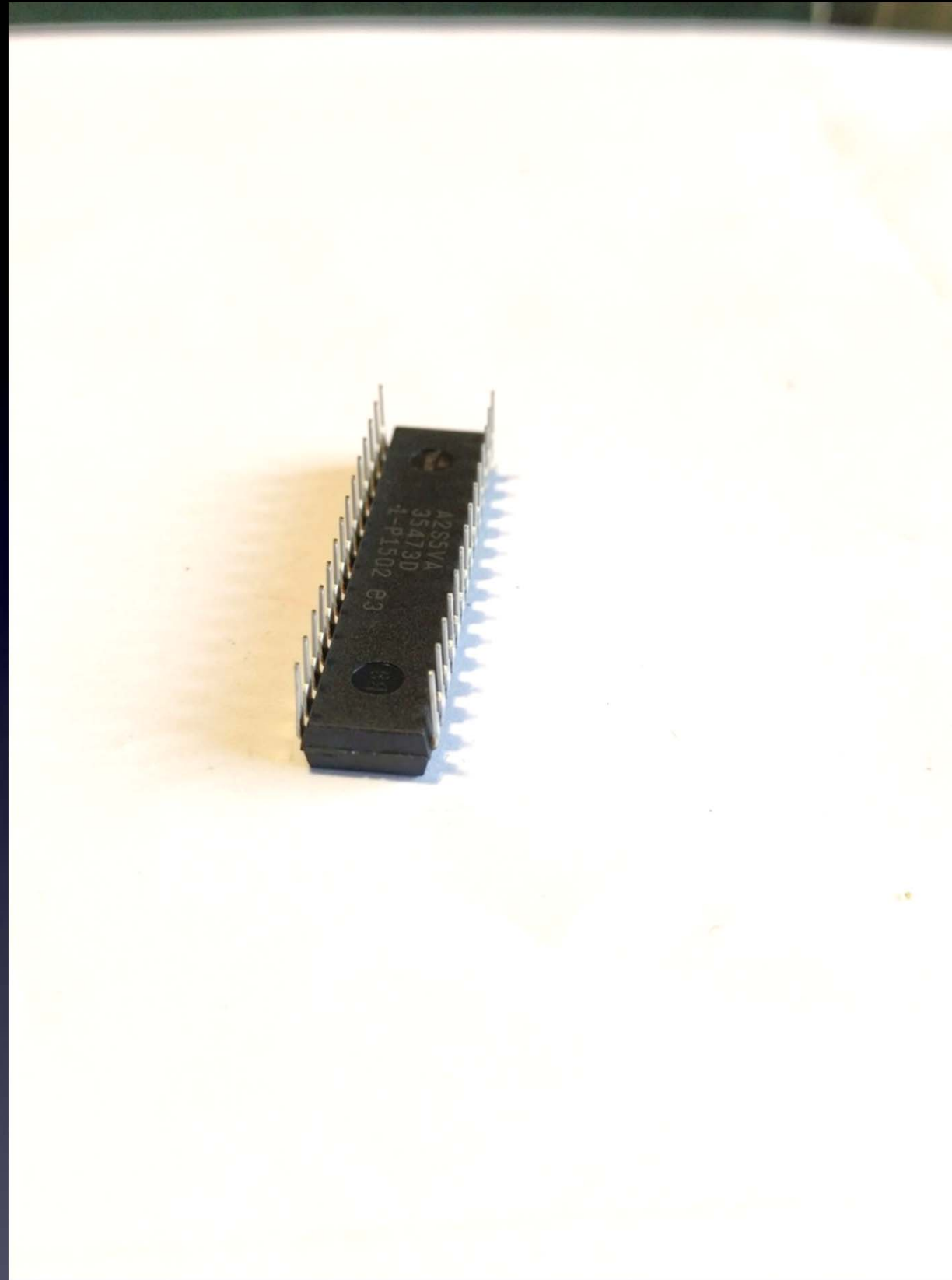
Note: Your kit's U1 chip may or may not have its pins already bent straight and parallel.
If not, you need to bend them, as shown in the next picture.

U1

Note: Your kit's U1 chip may or may not have its pins already bent straight and parallel.
If not, you need to bend them, as shown in this picture.

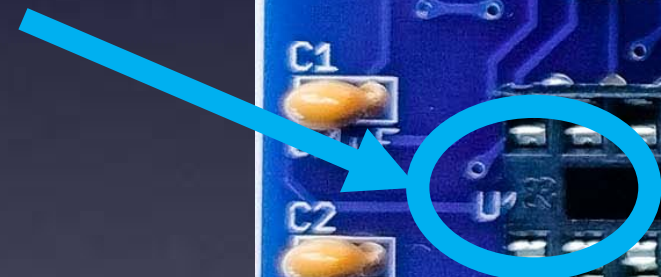


**We need the pins bent straight and parallel.
Use your work table to (gently) bend the leads.**

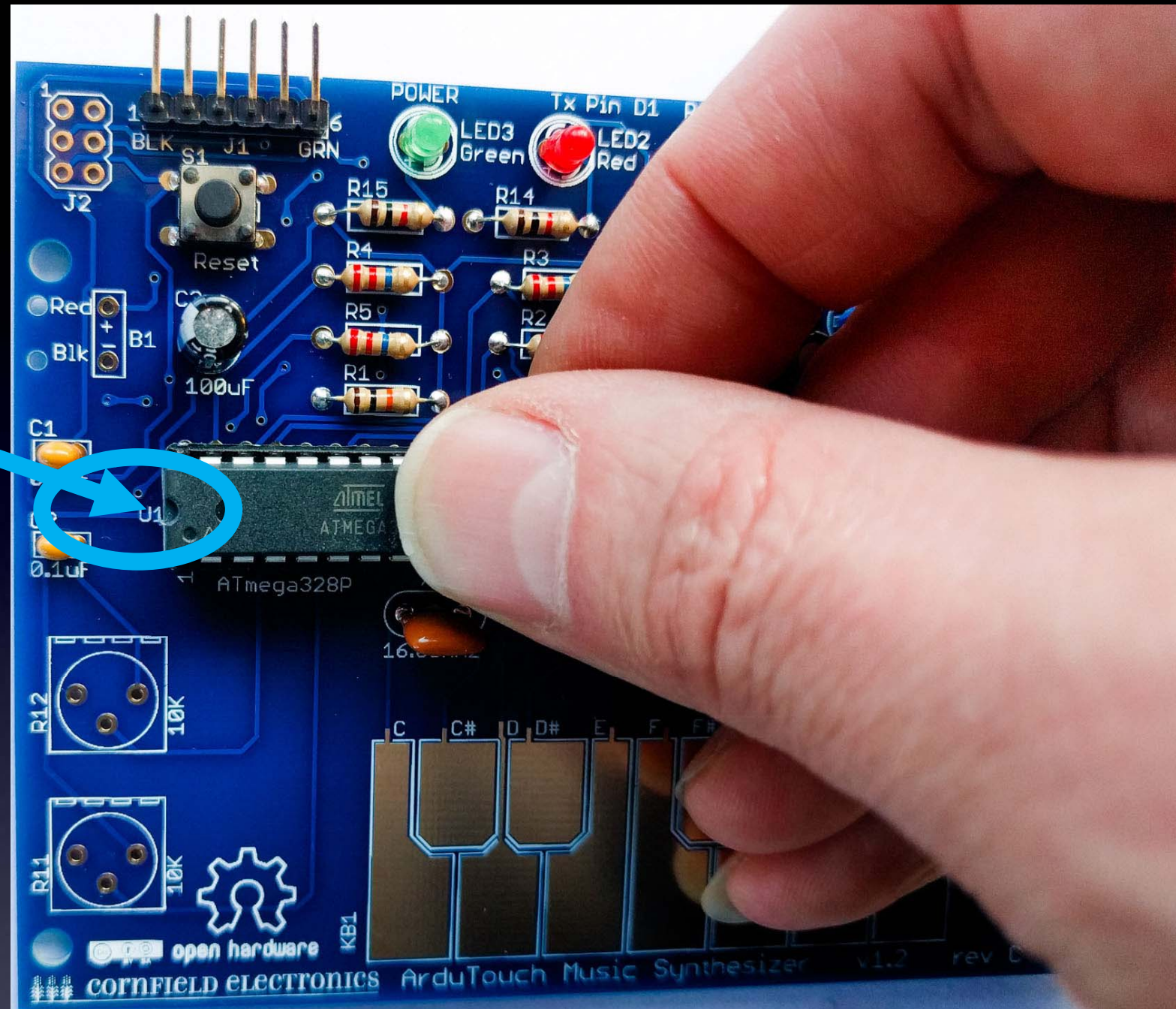


U1: microcontroller

These pins must be straight and parallel



U1: microcontroller



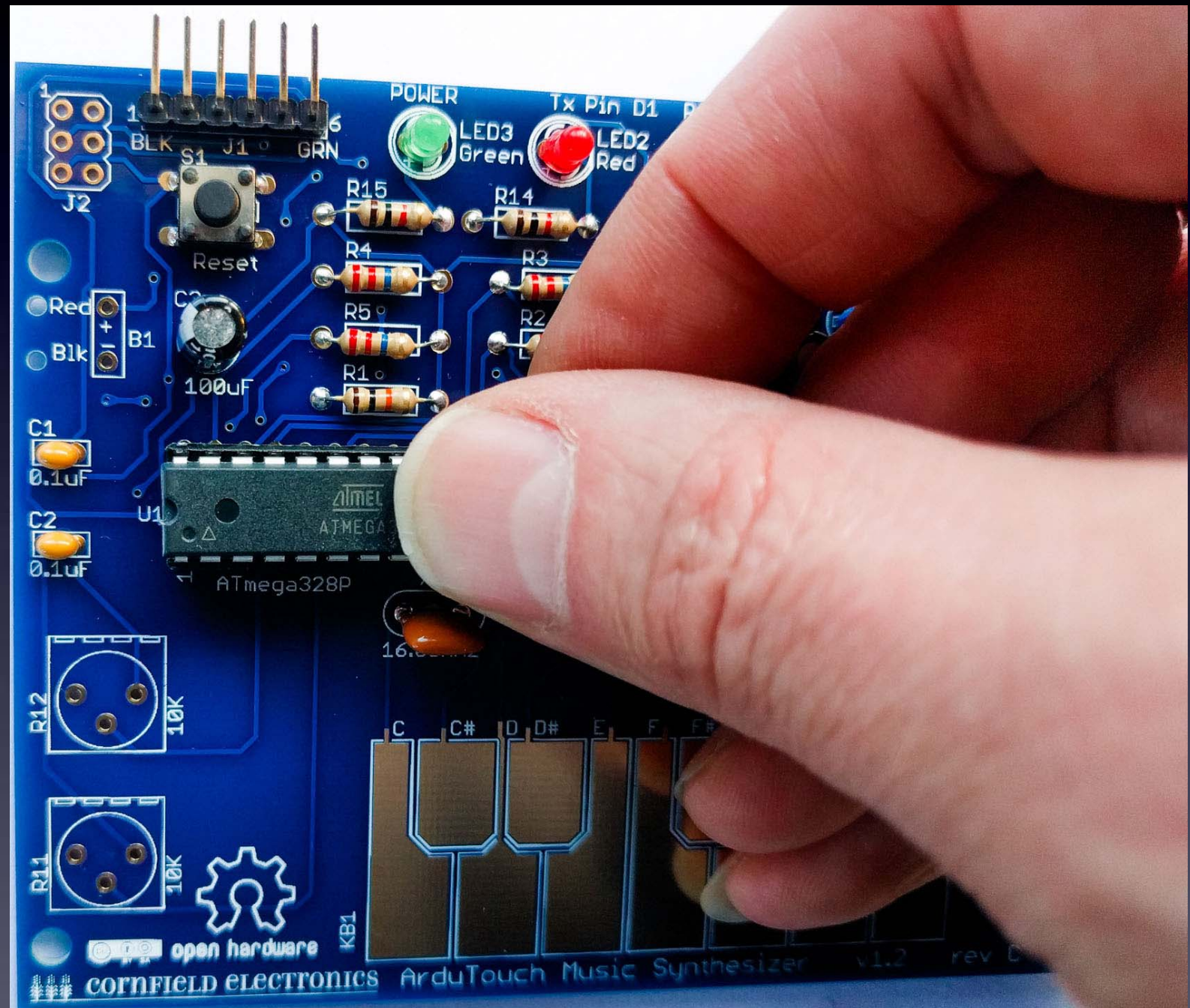
U1: microcontroller

make sure each pins rests in its hole in the socket
→ with the proper orientation

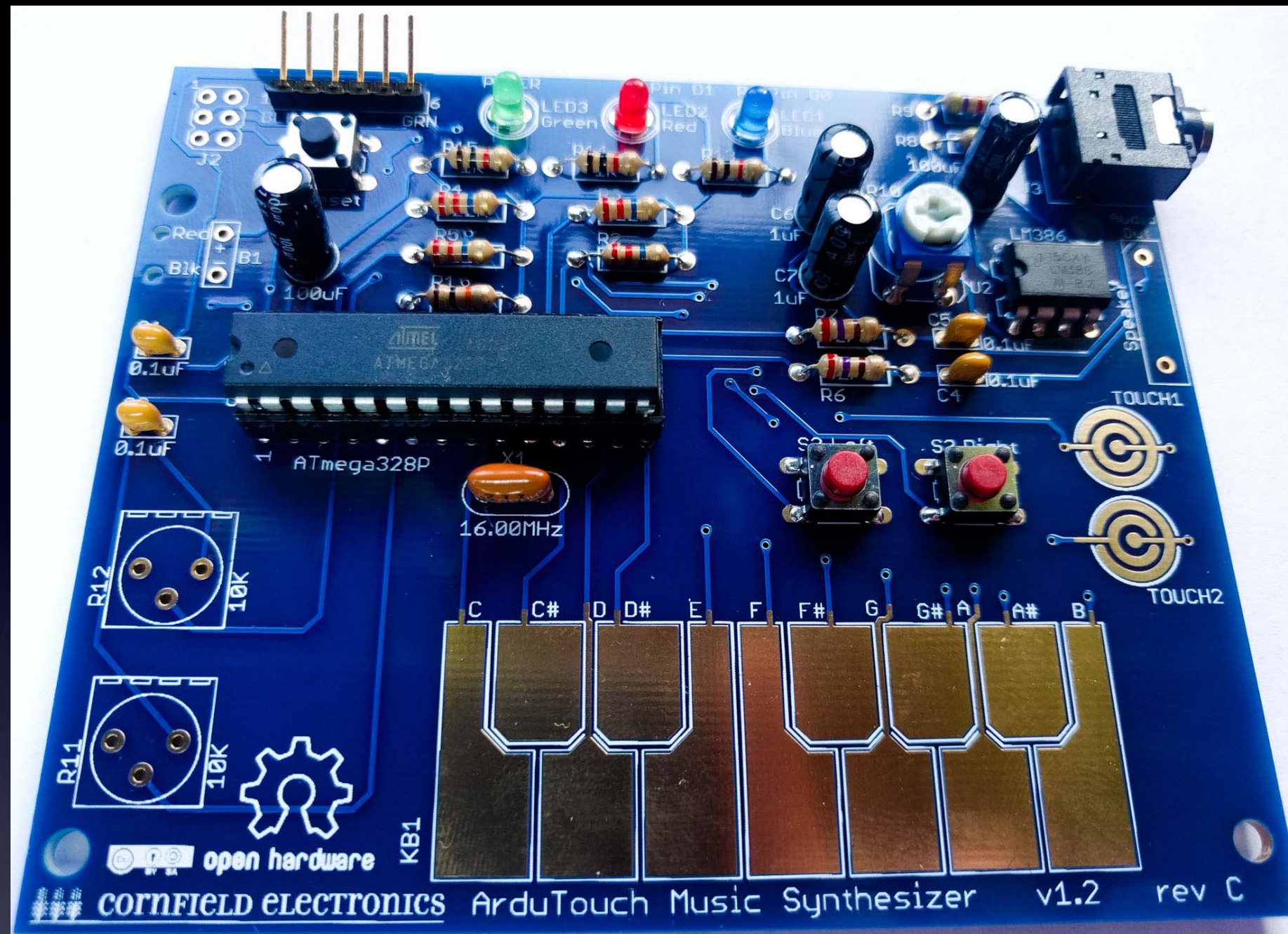
Use two thumbs to push microcontroller into its socket

**Make sure all 28 pins
are in place,
and push it into its socket.**

**(This is actually way easier
with *2 thumbs*.)**



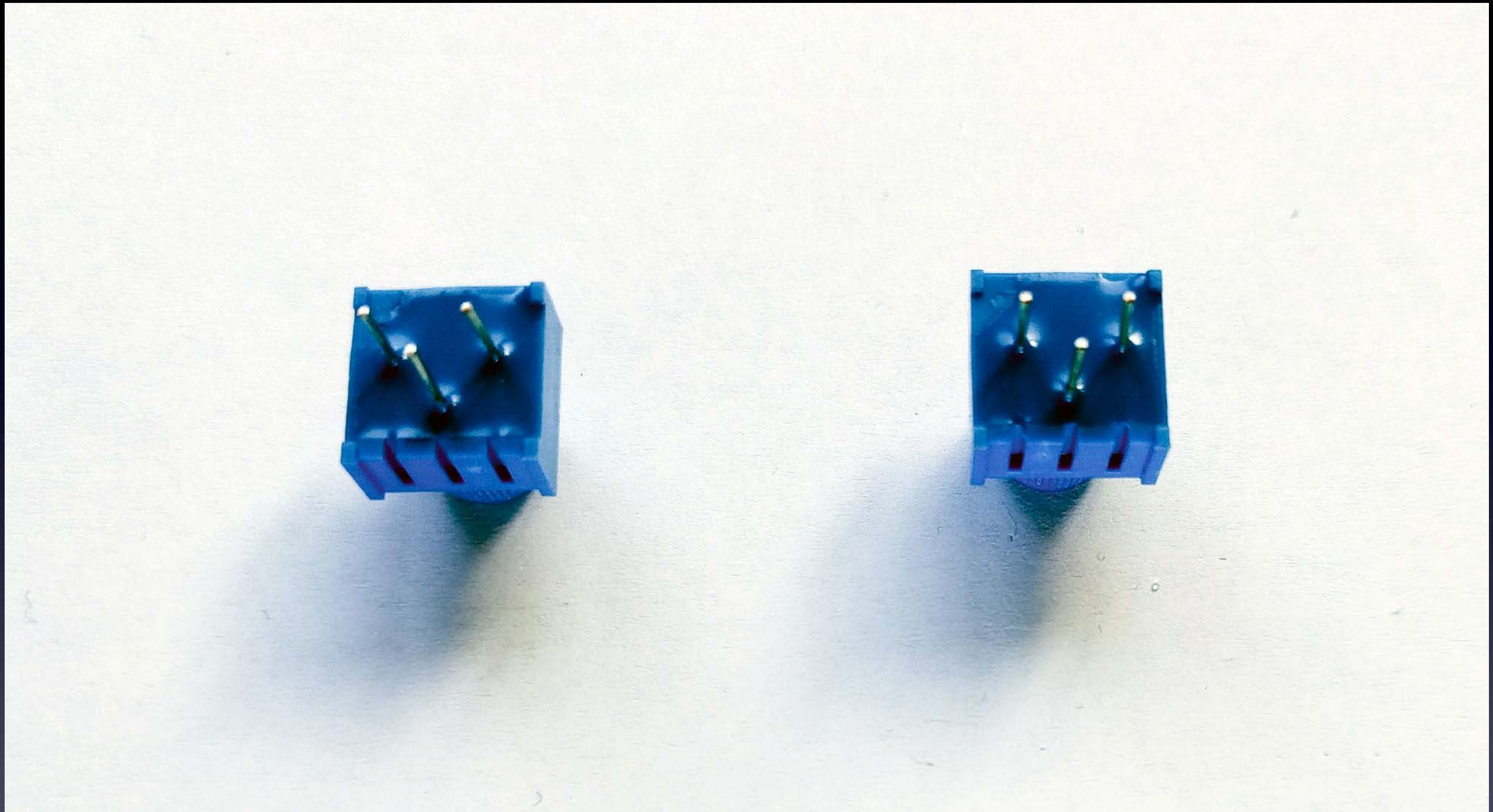
U1: microcontroller



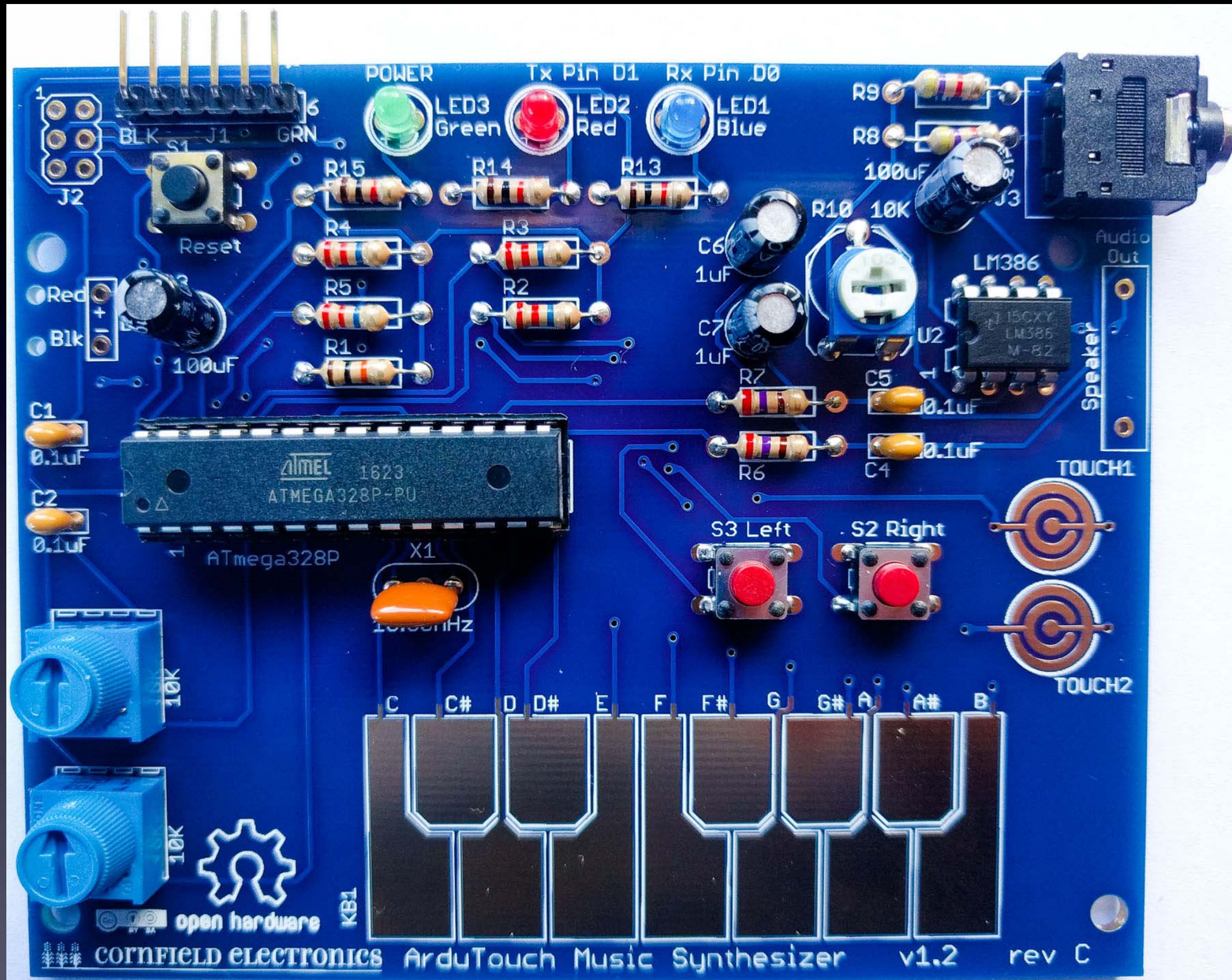
U1: microcontroller

Inspect all pins, and be sure each went into its hole in the socket – not bent.

If any pins are bent, (gently) pry out chip, straighten pins, and insert again.



R11 & R12: potentiometers

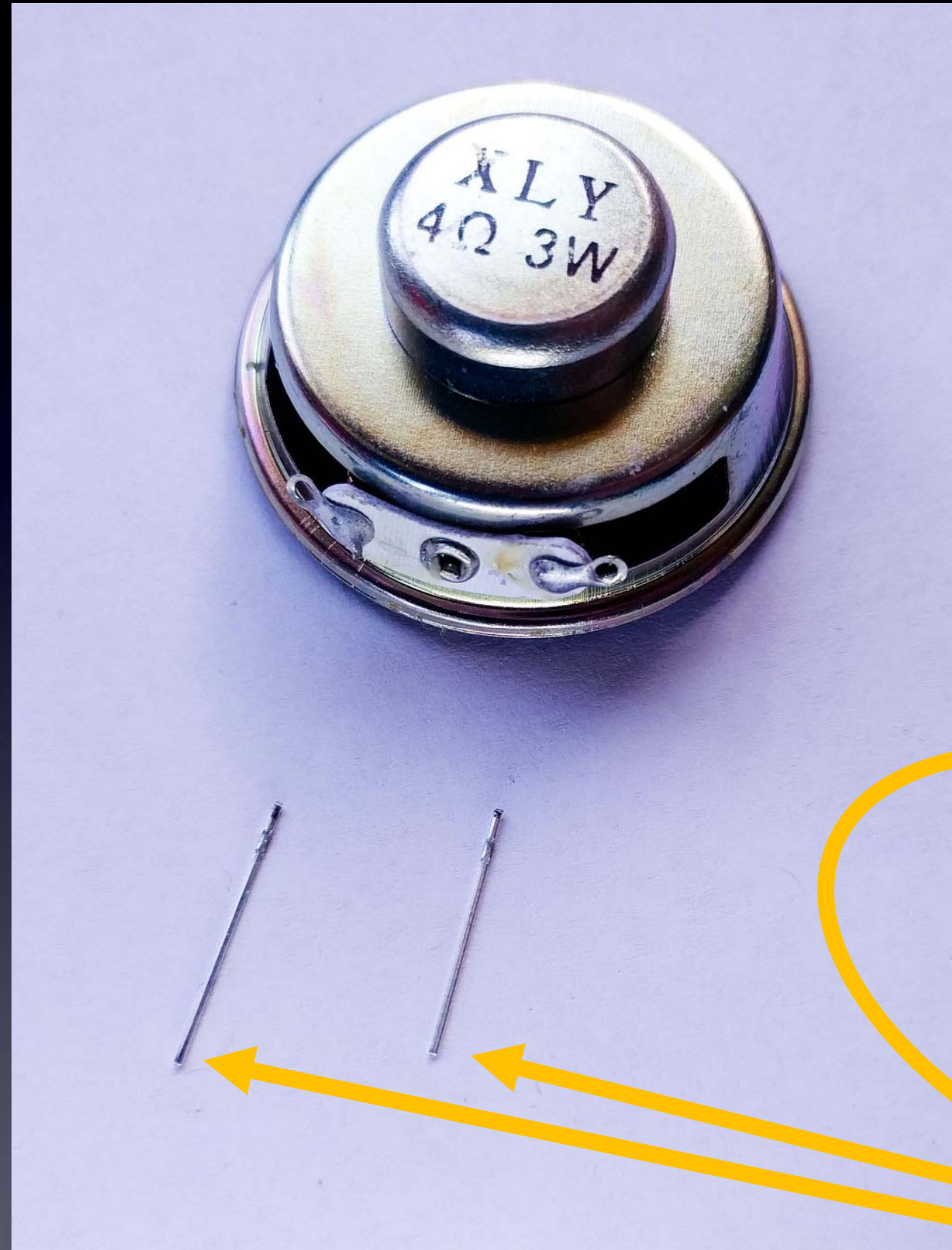


R11 & R12: potentiometers



Speaker

We'll add leads
to the speaker

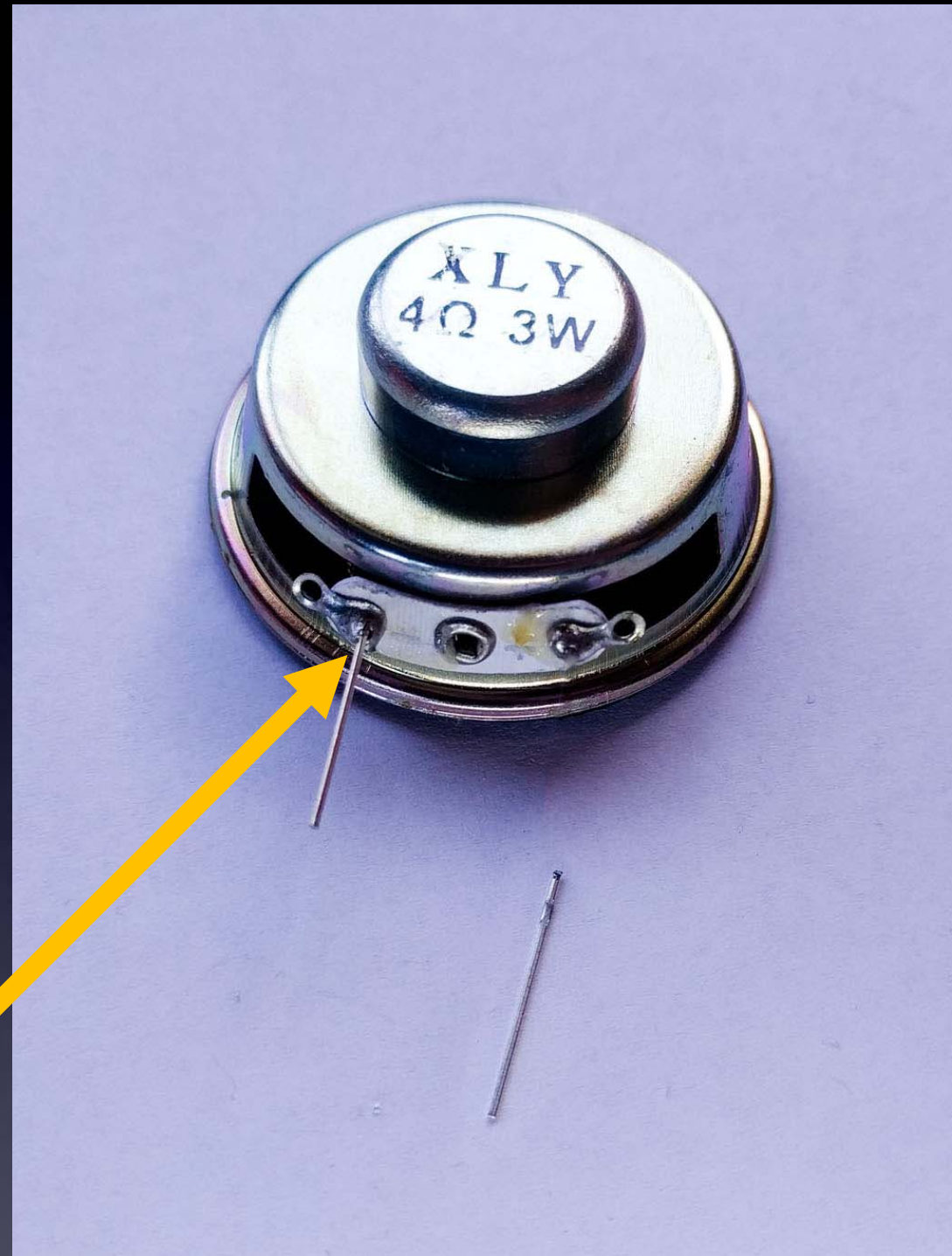


Saved
leads

from the LEDs

Speaker

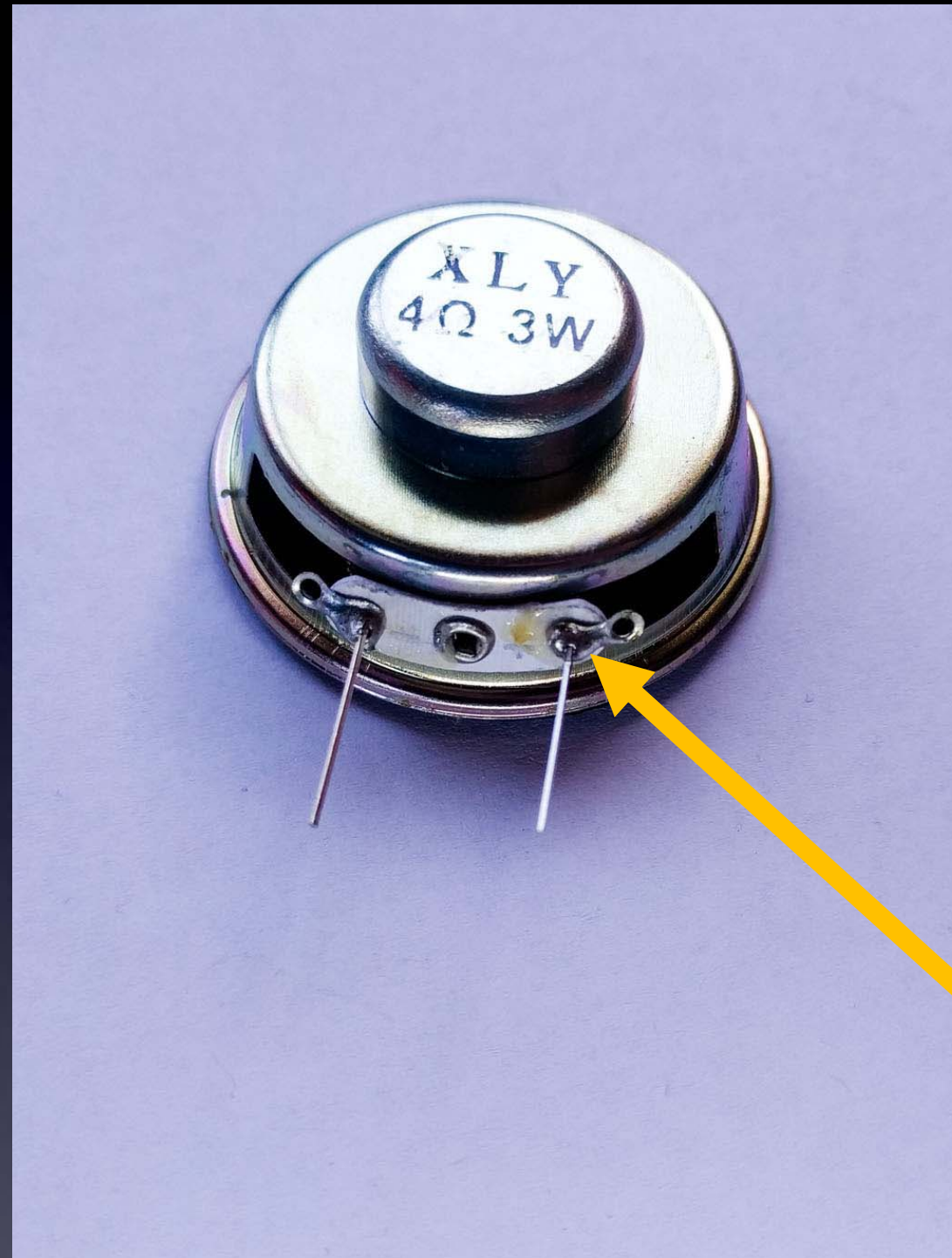
**Solder one lead
to speaker**



**Notice the
correct place
to solder the wire**

Speaker

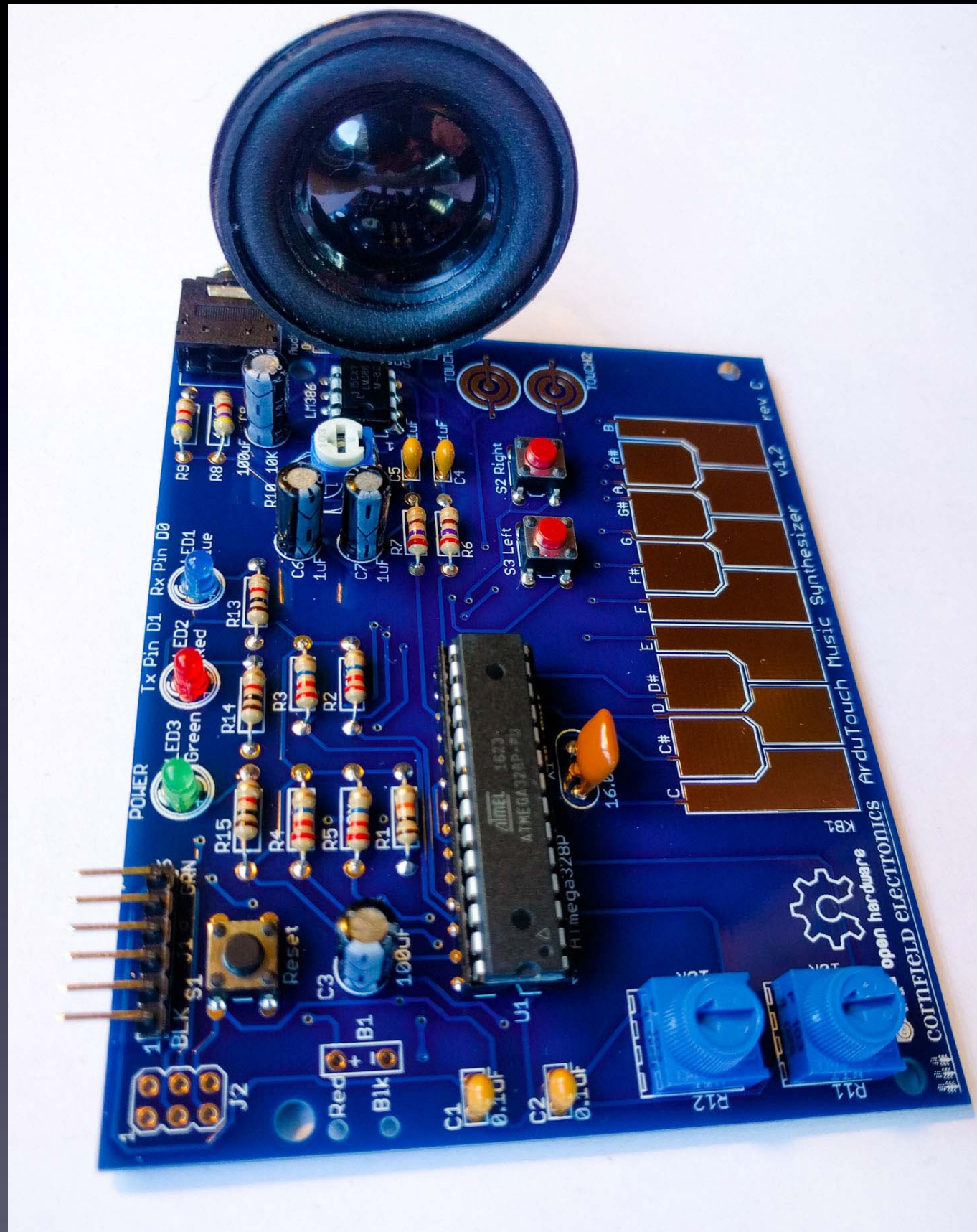
**Solder next lead
to speaker**



**Notice the
correct place
to solder the wire**

Speaker

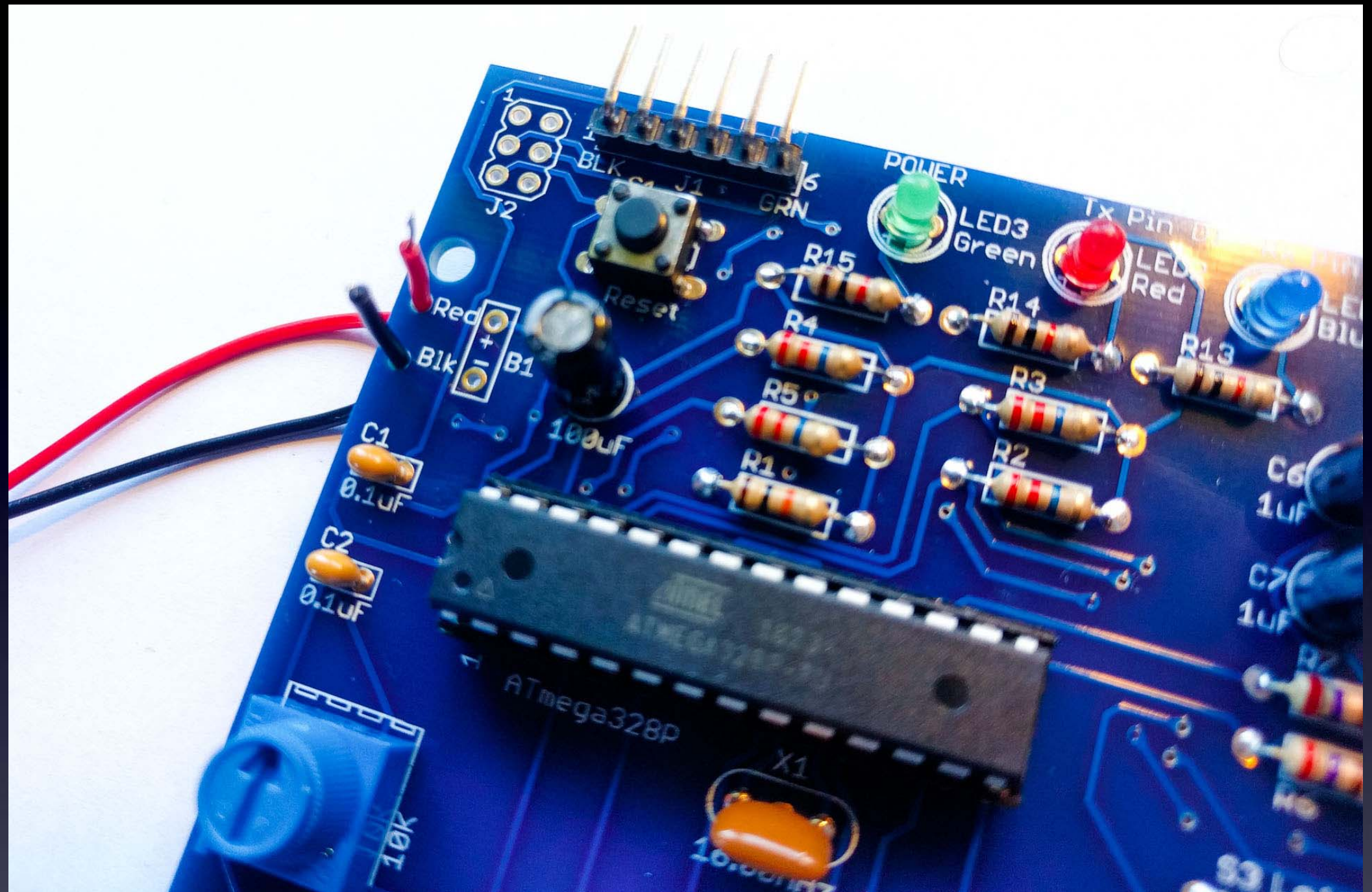
**Insert
speaker into board
and solder
both leads to board.**



Speaker

Note: Some battery pack wires have thicker red and black plastic coatings.

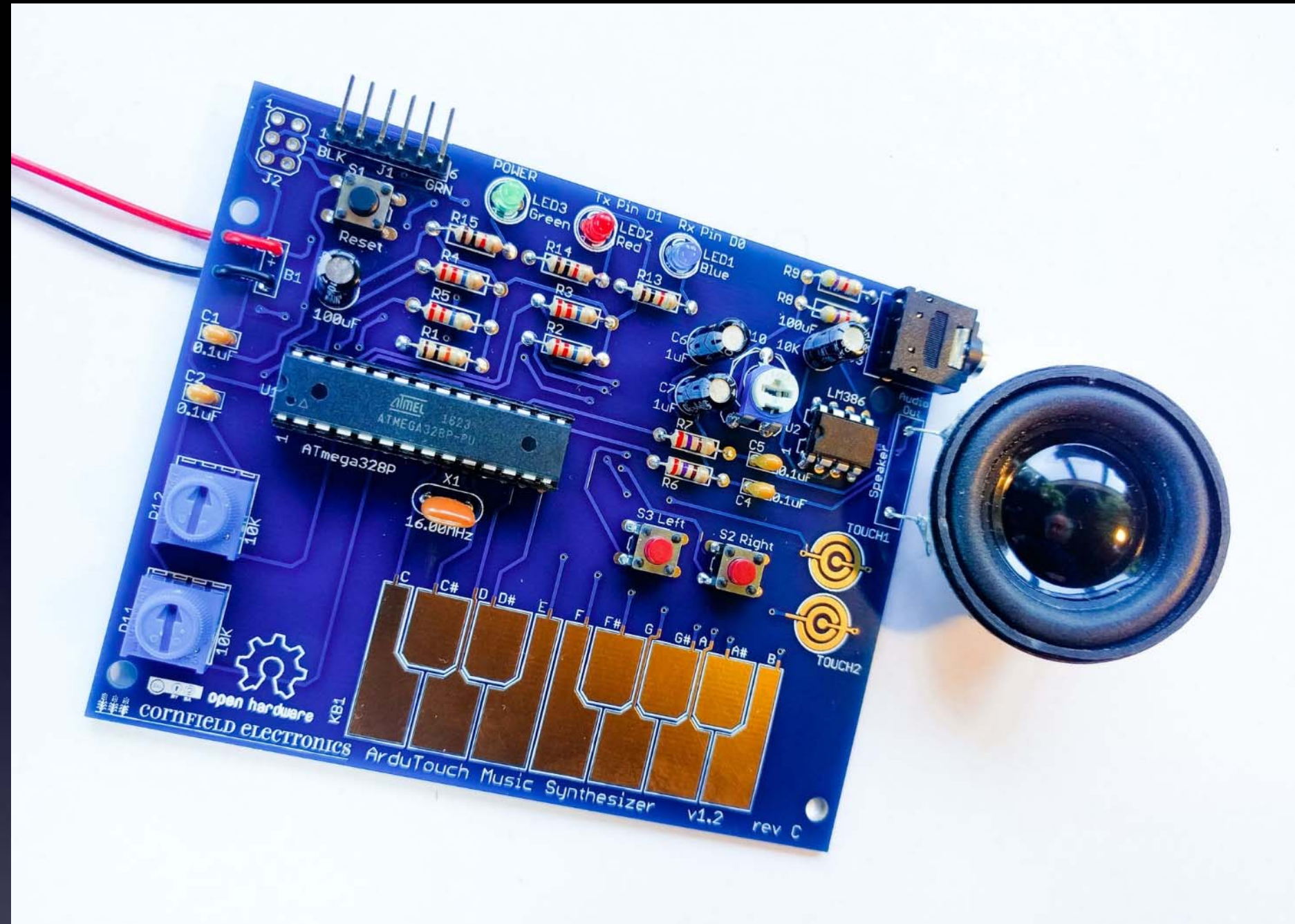
If so, you can widen the these two holes by gently rotating a scissors or small knife or small Phillips screwdriver on the top and bottom of these two holes.



Push battery pack leads through holes.

Make sure Red and Black go through their correct holes!

Battery pack



Loop one lead into its pad,
and solder.

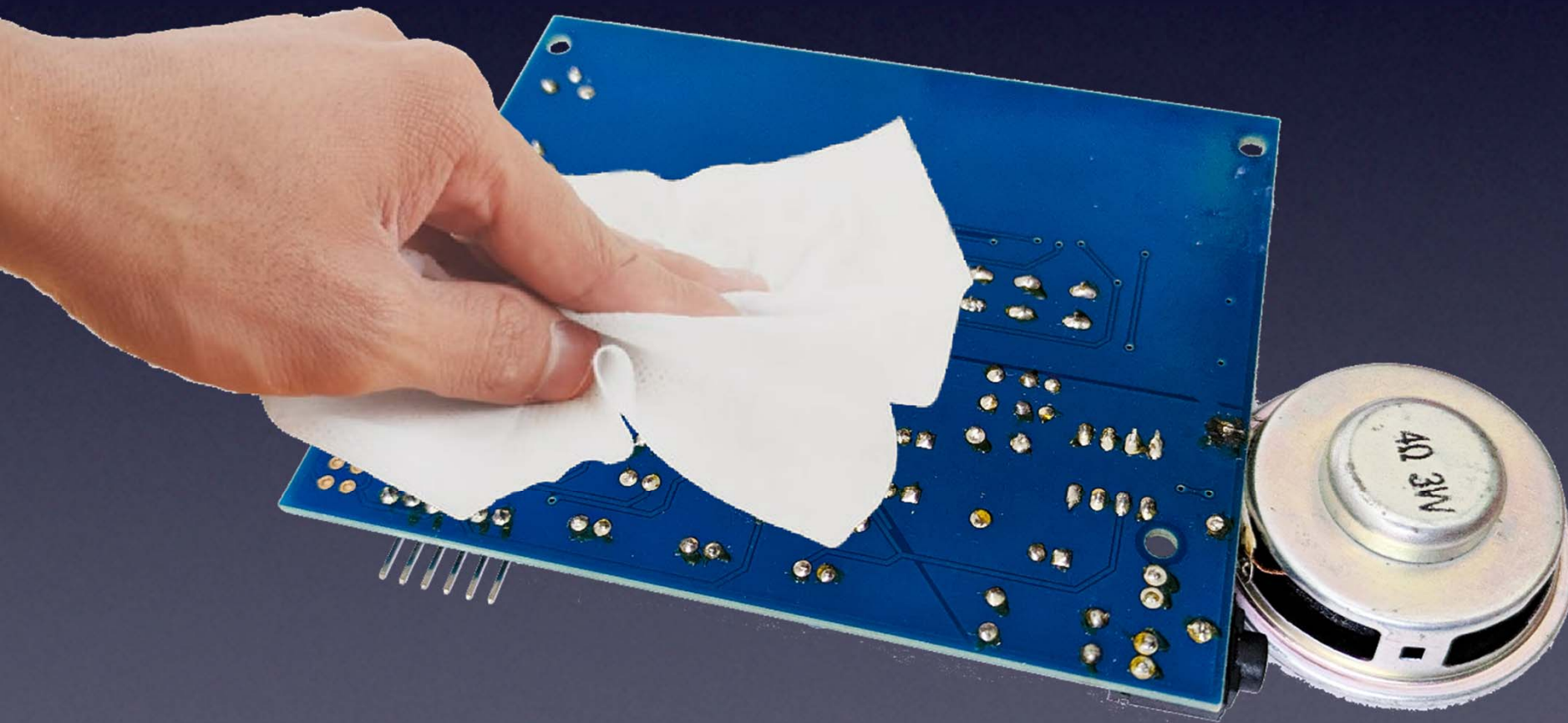
Then loop the other lead into its pad,
and solder.

Battery pack

If you used any *flux paste* for re-working problems

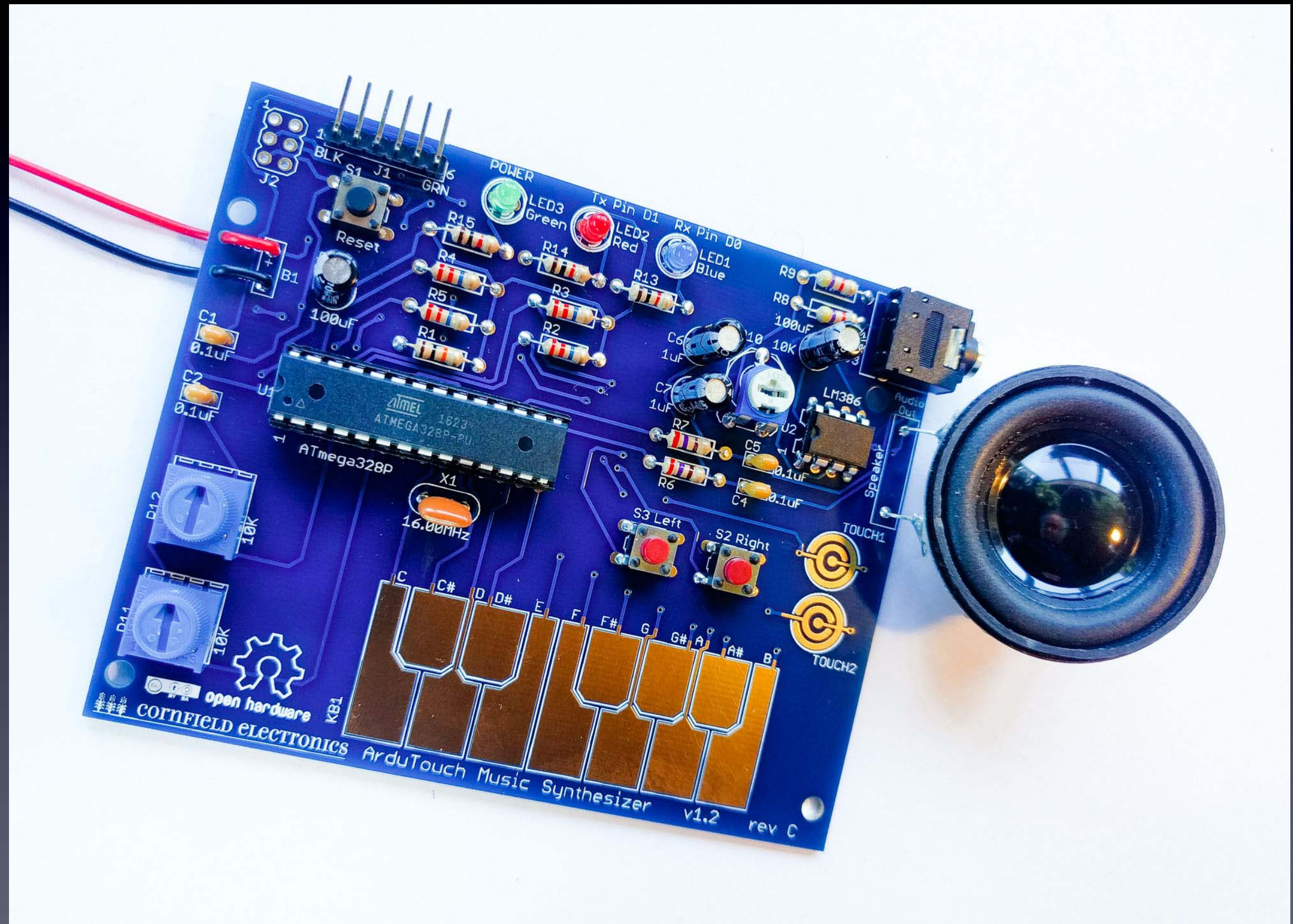


The bottom of the PCB will be sticky from the flux

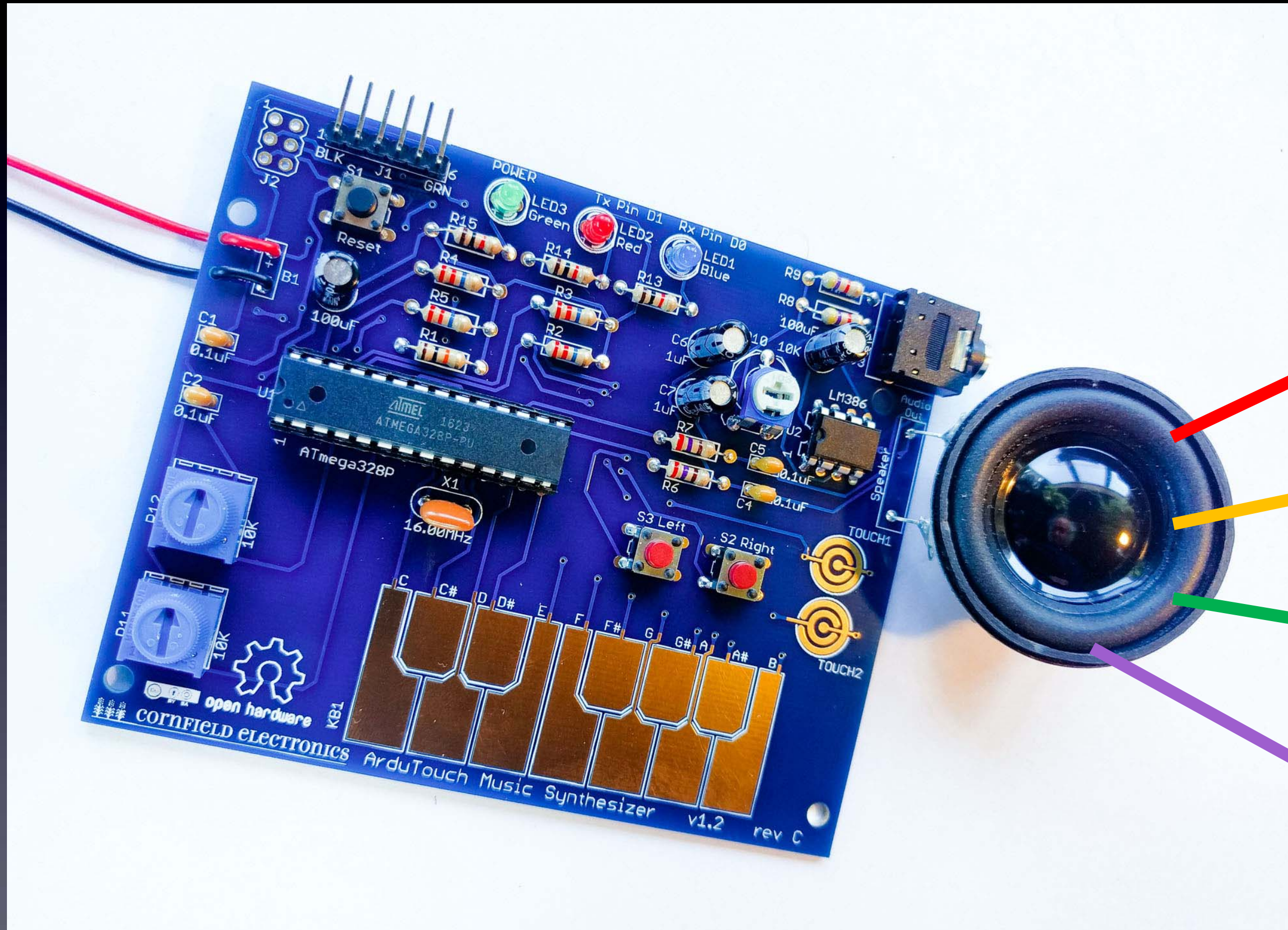


You can clean it with a cloth wet with *Isopropyl Alcohol*

Done!



Let's make noise!



Please Remember:

to

Wash your hands

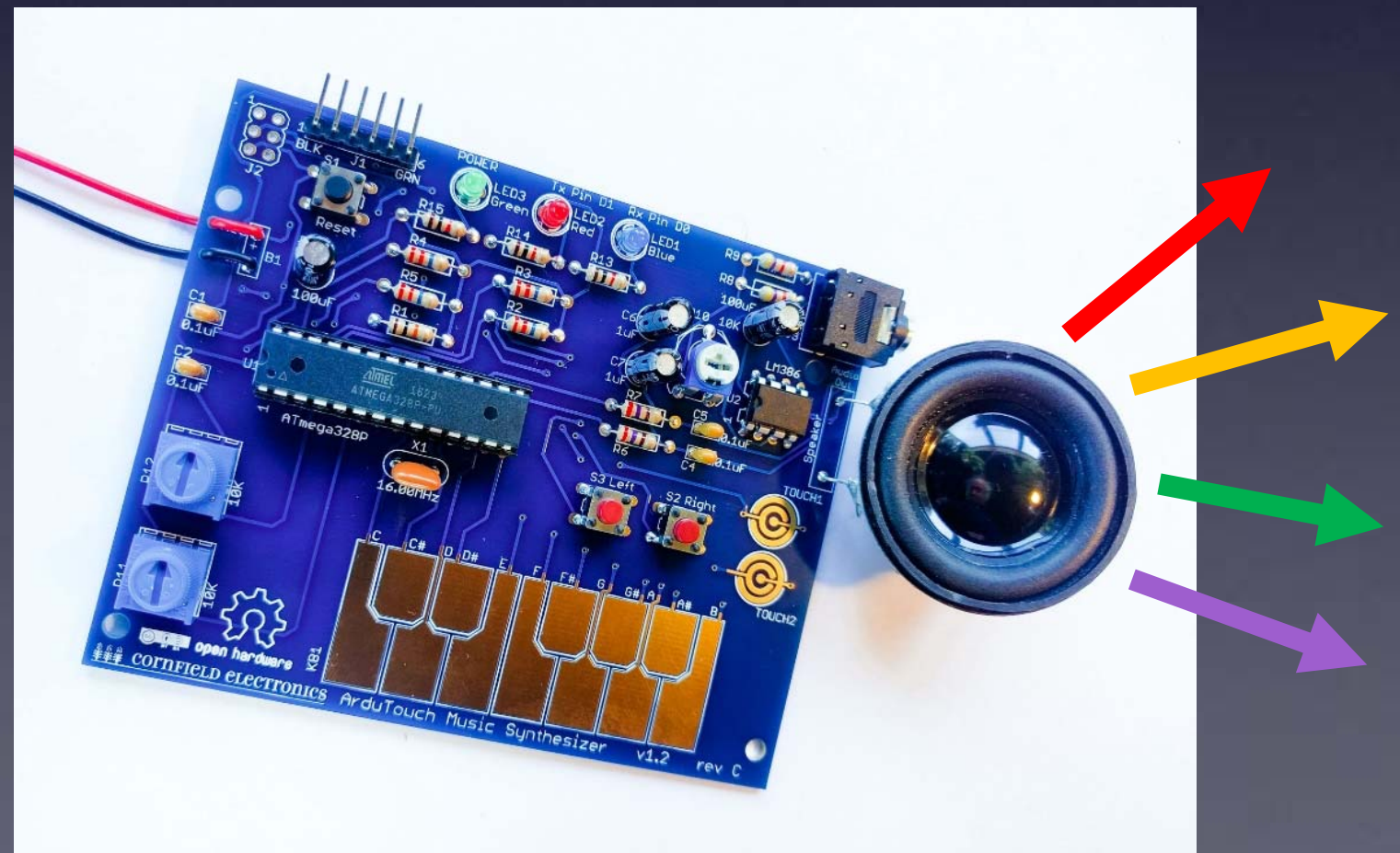
after soldering

Let's make noise!

Your ArduTouch comes pre-programmed with a really cool synthesizer, called “Thick”.

“Thick” plays 4 sawtooth waveform notes at once.

- the left and right buttons change octaves
- long press the left and right buttons to change sounds
- the Bottom knob controls the glide rate
- the Top knob controls how each of the 4 notes glide separately
- Try playing with these and see!

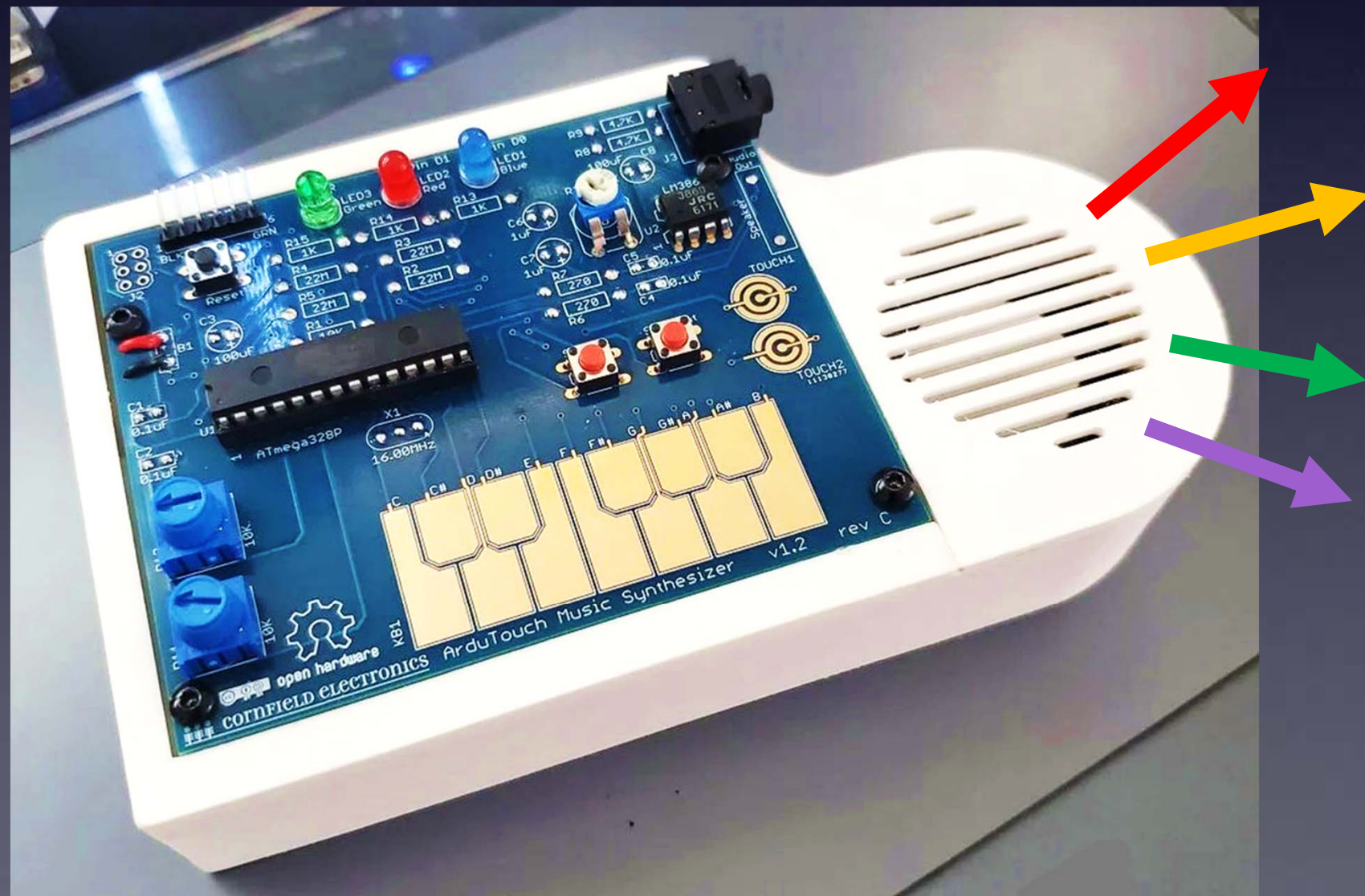


Optional: 3D Printed Case !

The built-in speaker will sound ****way**** nicer with a case.

ArduTouch Case, by ipsofatso – on Thingiverse:

<https://www.thingiverse.com/thing:4702927>

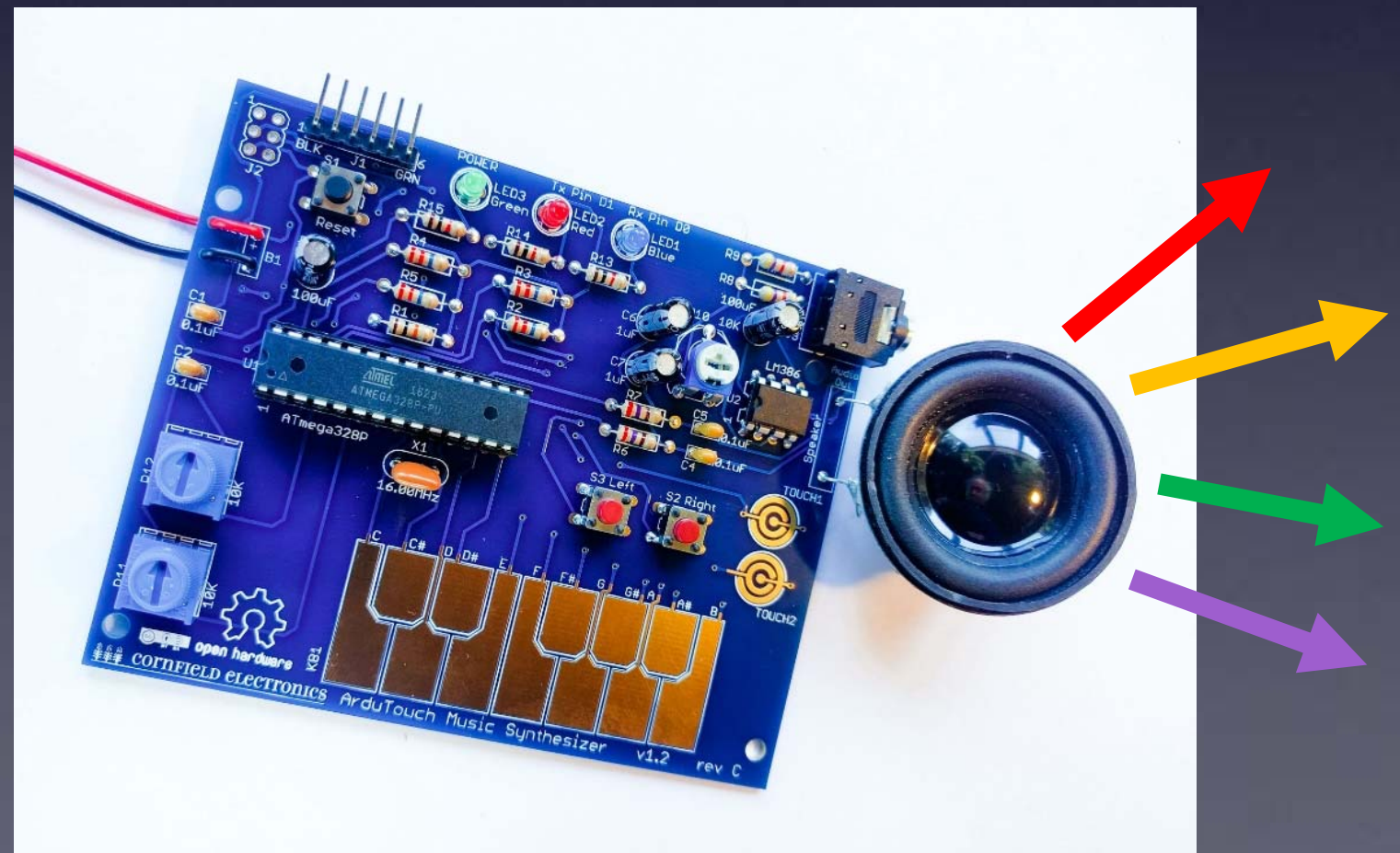


Let's make noise!

Your ArduTouch comes pre-programmed with a really cool synthesizer, called “Thick”.

If you are happy playing with “Thick” then no need to re-program your ArduTouch.

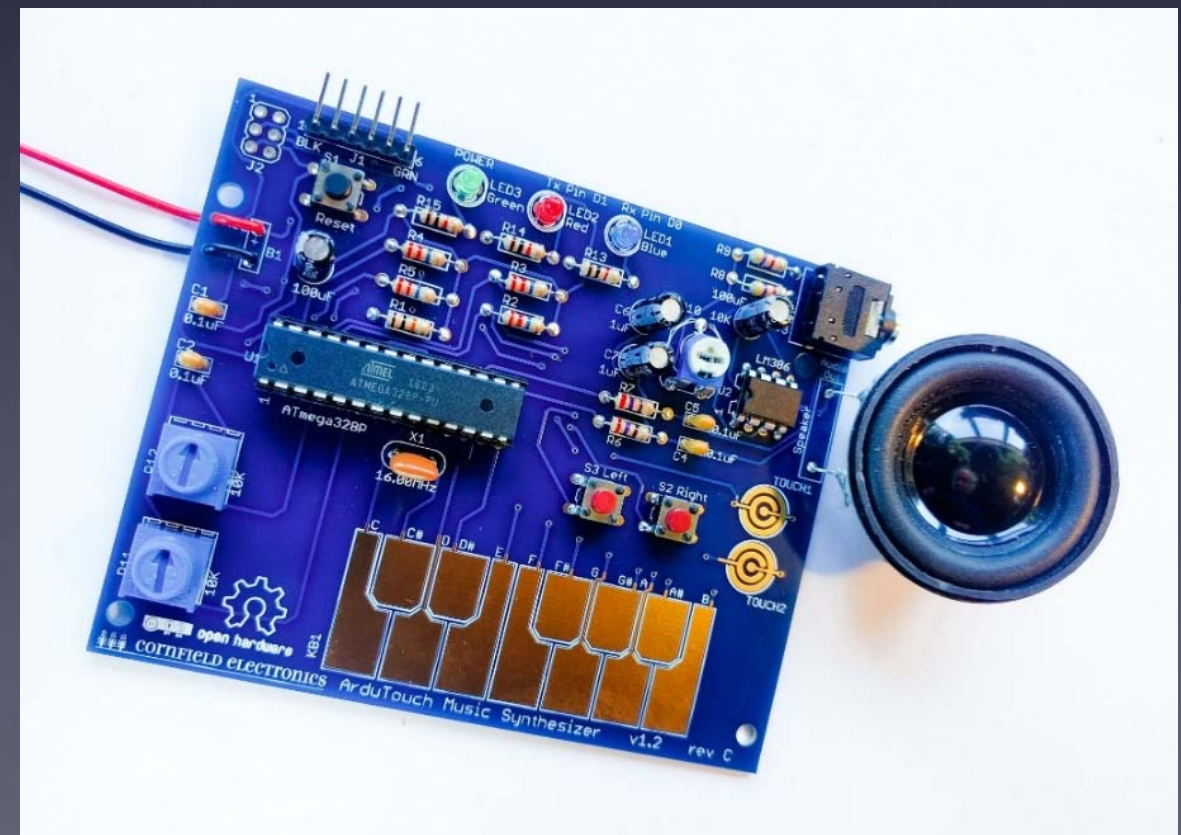
But if you want to program other synths into your ArduTouch, the next pages show you how...



Re-programming the ArduTouch

**We have written several way cool synthesizers for the ArduTouch!
Each is unique, and each way different than the others.**

The following slides show you
how to program these into your ArduTouch board...



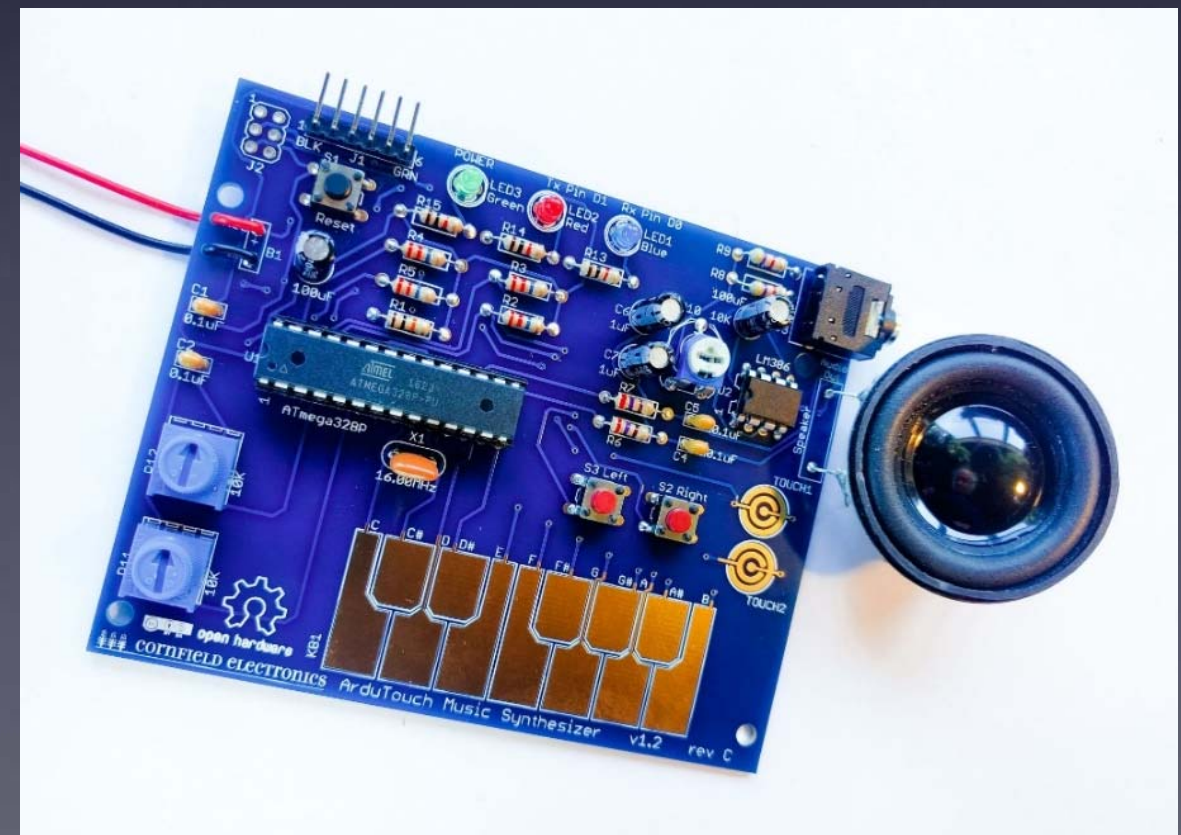
Re-programming the ArduTouch

**We have written several way cool synthesizers for the ArduTouch!
Each is unique, and each way different than the others.**

To program in a new synth in your ArduTouch, you will need:

- the Arduino software
<<http://arduino.cc>>
- a USB-Serial adapter cable (such as an FTDI, or equivalent)
a nice one is available at
<<https://cornfeldelectronics.com/cfe/products/buy.php?productId=usbcable>>
- a synth sketch & the ArduTouch Arduino library
<<http://cornfeldelectronics.com/cfe/projects.php#ardutouch>>

**The following slides show you
how to do the above, in detail.**



Arduino

Arduino is a very powerful tool!

But it is very easy to use.

It was designed for total beginners to use successfully.

I won't give a complete tutorial here – just some basics.

For more info, there are many good Arduino tutorials online.

A good place to start is:

<<https://www.arduino.cc/en/Tutorial/HomePage>>



Intro to Arduino



Arduino For Total Newbies workshop

Day 3 Tuesday 29-December, 13:00 to 16:30

→ → *Right-click on this link, and open in a new window*
[Arduino For Total Newbies workshop room on Big Blue](#)

NOTE: You do NOT need to register to take this works
Just show up before the start time at the Big Blue Button room,
given above.

*Learn Arduino
using TV-B-Gone
as an example project
(no materials required)*



Arduino For Total Newbies workshops

Arduino

First:

Download and install the Arduino software

< <http://arduino.cc> >

Any version is OK

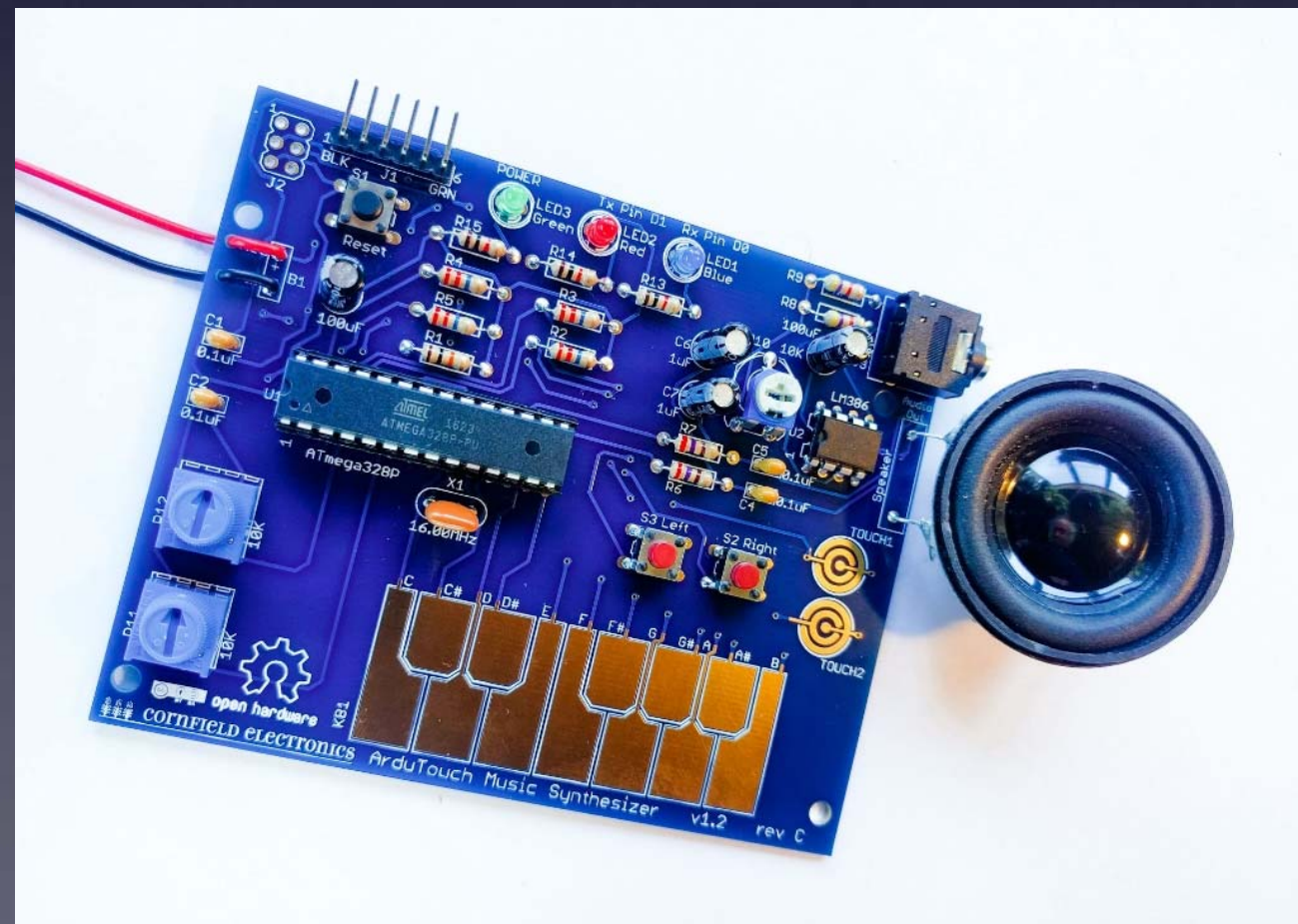


Re-programming the ArduTouch

Second:

Download and install the ArduTouch Arduino library
<<http://cornfieldelectronics.com/cfe/projects.php#ardutouch>>

(details on this soon)



Re-programming the ArduTouch

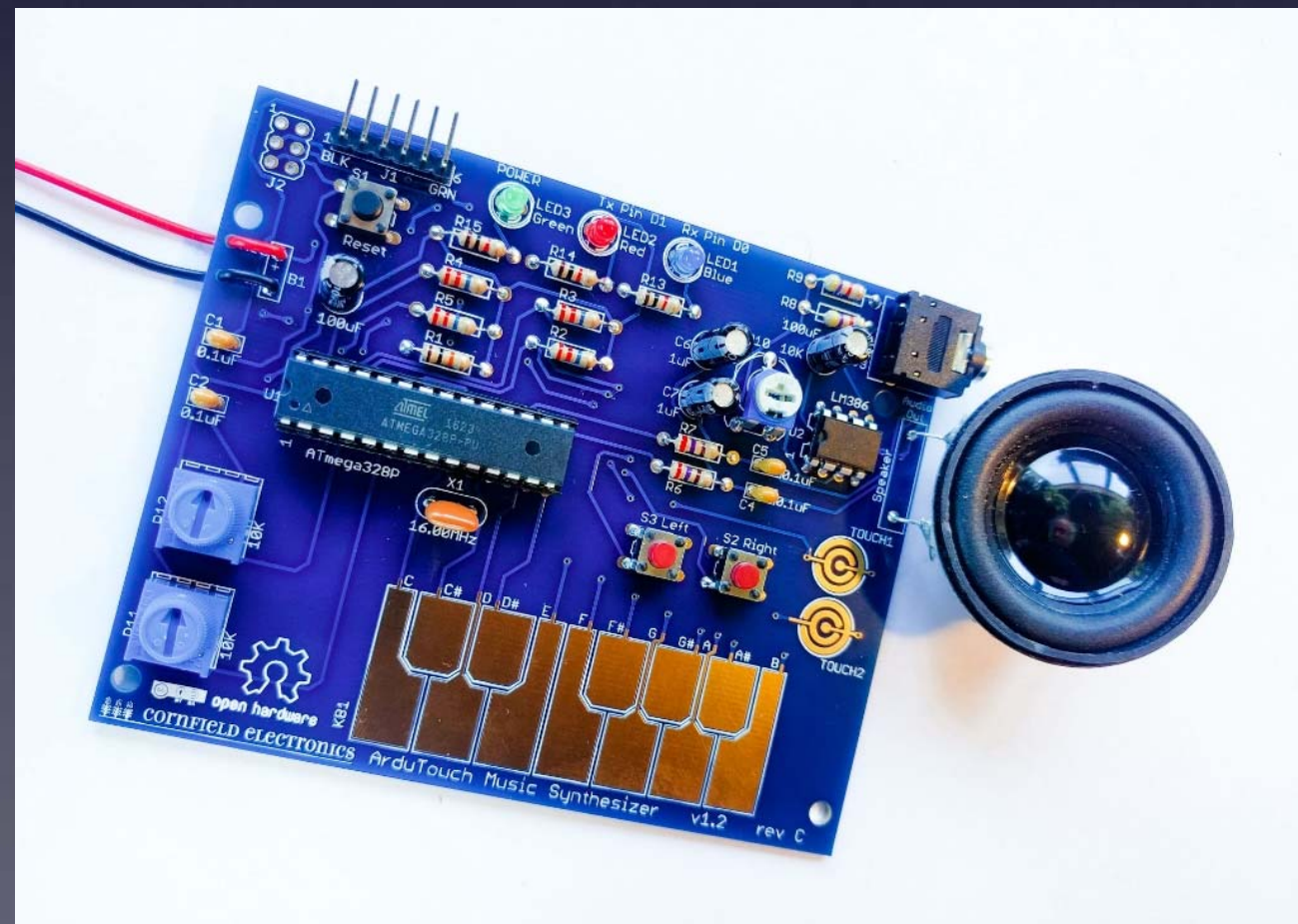
Third:

Download ArduTouch synth sketches

[<http://cornfieldelectronics.com/cfe/projects.php#ardutouch>](http://cornfieldelectronics.com/cfe/projects.php#ardutouch)

Store them on your computer anywhere you like.

(details on this soon)

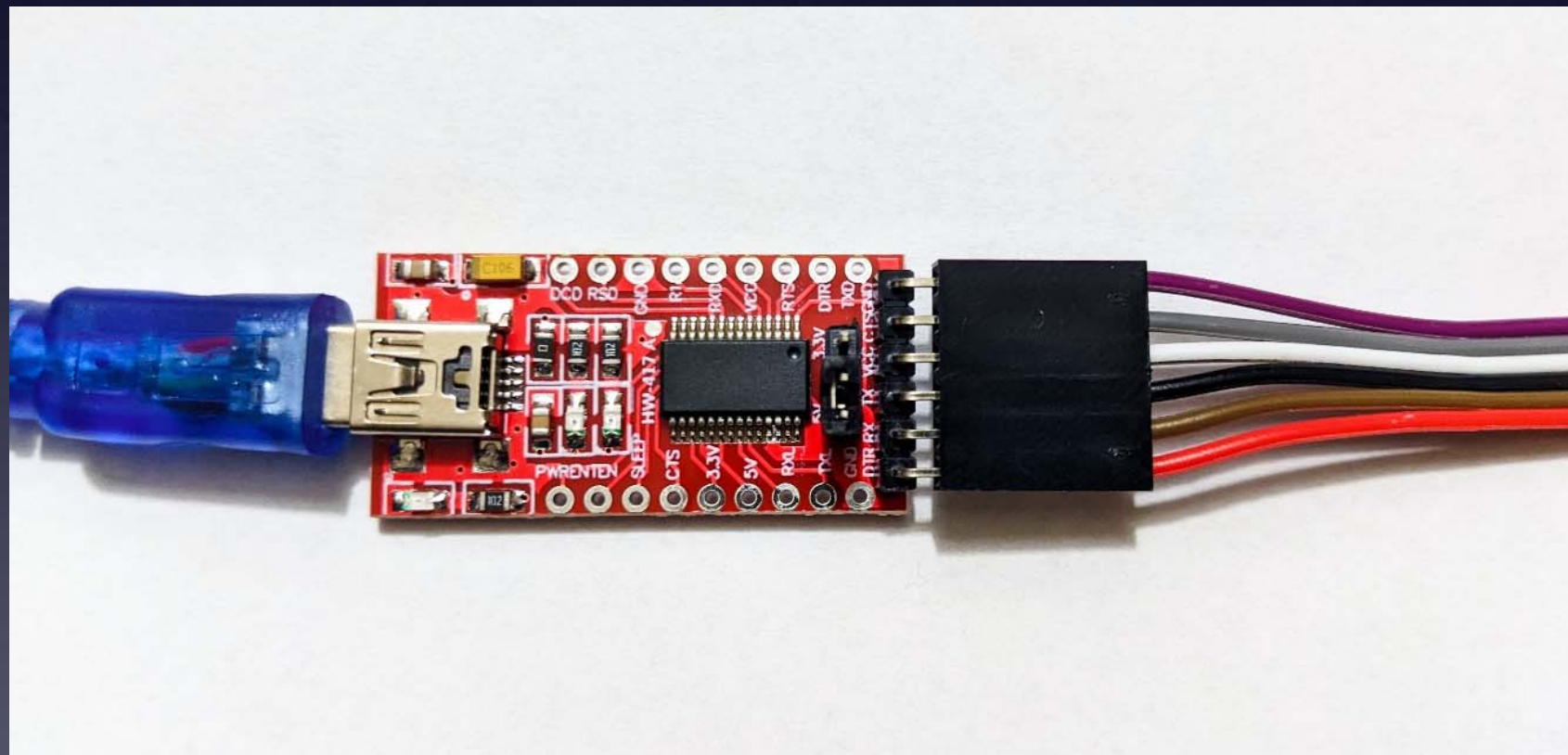


Connecting your ArduTouch to your computer

USB-Serial adapter cable

Ones available from Cornfield Electronics look like this:

[<https://cornfieldelectronics.com/cfe/products/buy.php?productId=usbcable>](https://cornfieldelectronics.com/cfe/products/buy.php?productId=usbcable)

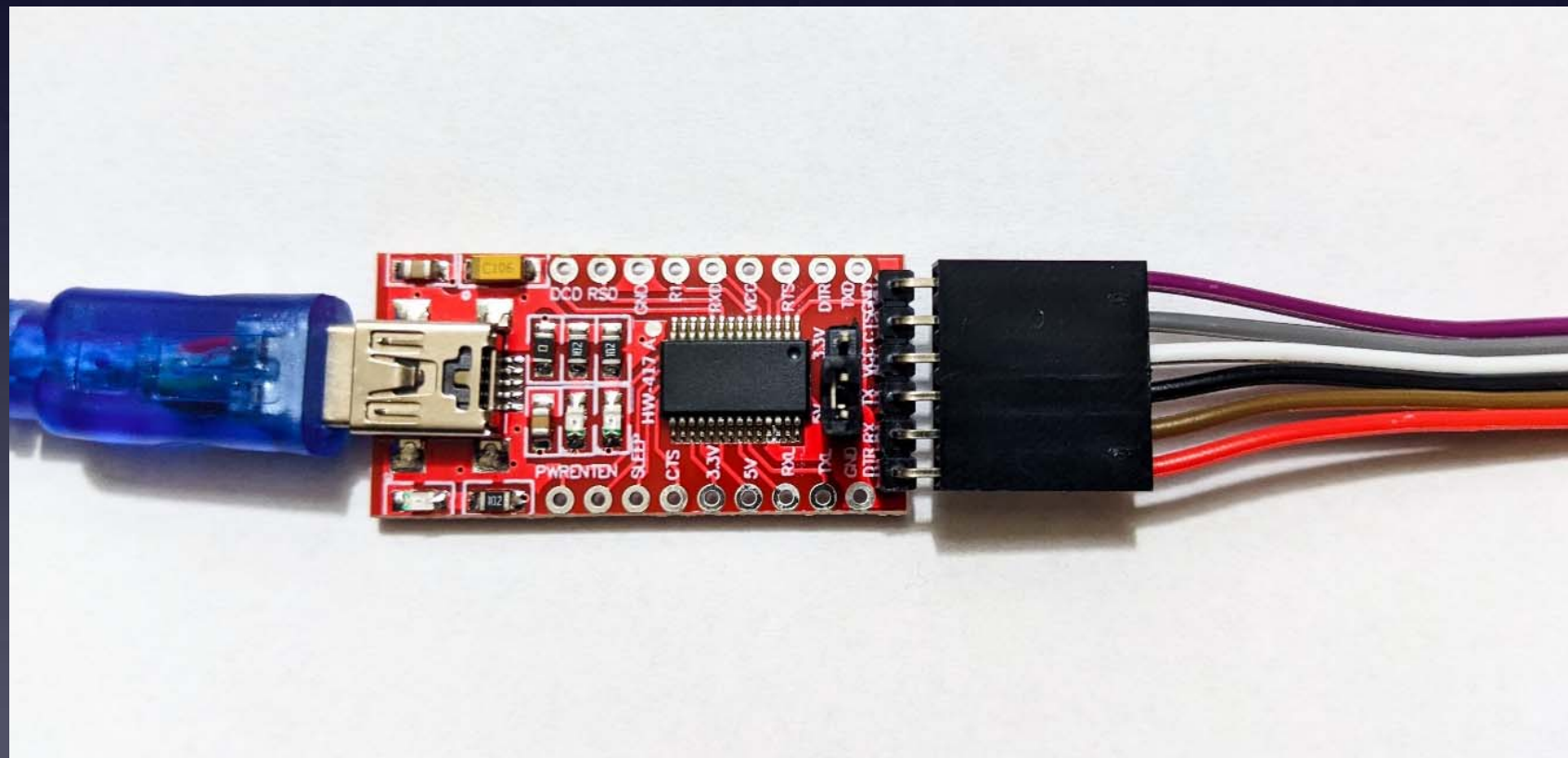


Connecting your ArduTouch to your computer

USB-Serial adapter cable

Ones available from Cornfield Electronics look like this:

[<https://cornfieldelectronics.com/cfe/products/buy.php?productId=usbcable>](https://cornfieldelectronics.com/cfe/products/buy.php?productId=usbcable)

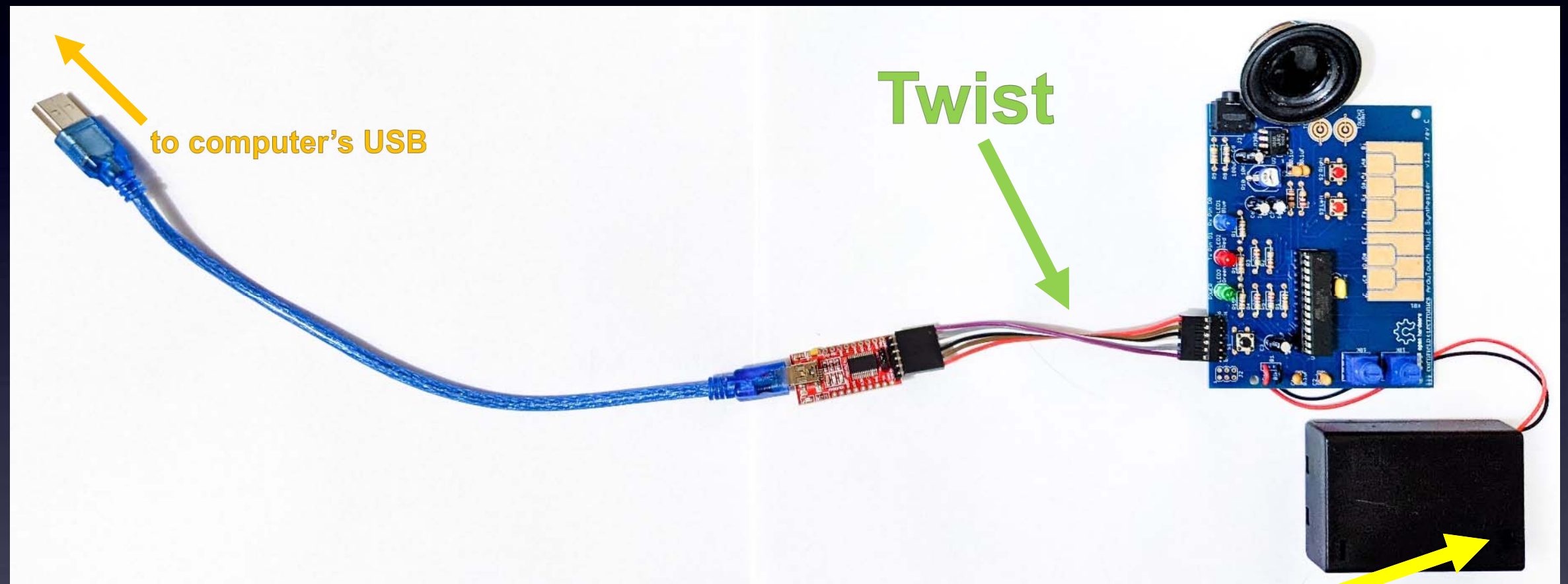


**You may need to download and install a driver
for your Operating System (Windows, MacOS, or Linux):**

[<https://ftdichip.com/drivers/vcp-drivers/>](https://ftdichip.com/drivers/vcp-drivers/)

Or search for:
“FTDI 232 driver”

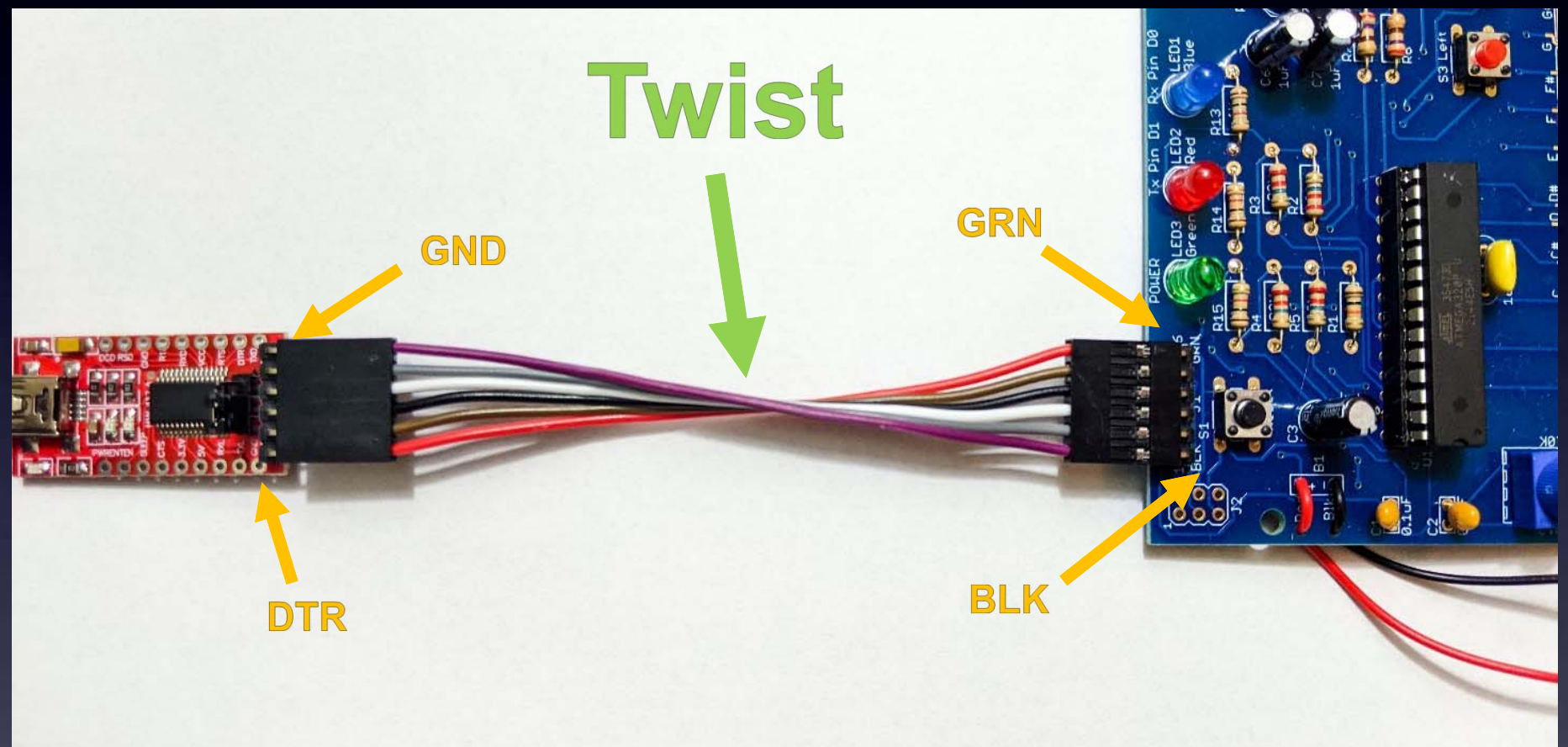
Connecting your ArduTouch to your computer



IMPORTANT:
Make sure the
battery pack on your
ArduTouch
is *OFF*

Connecting your ArduTouch to your computer

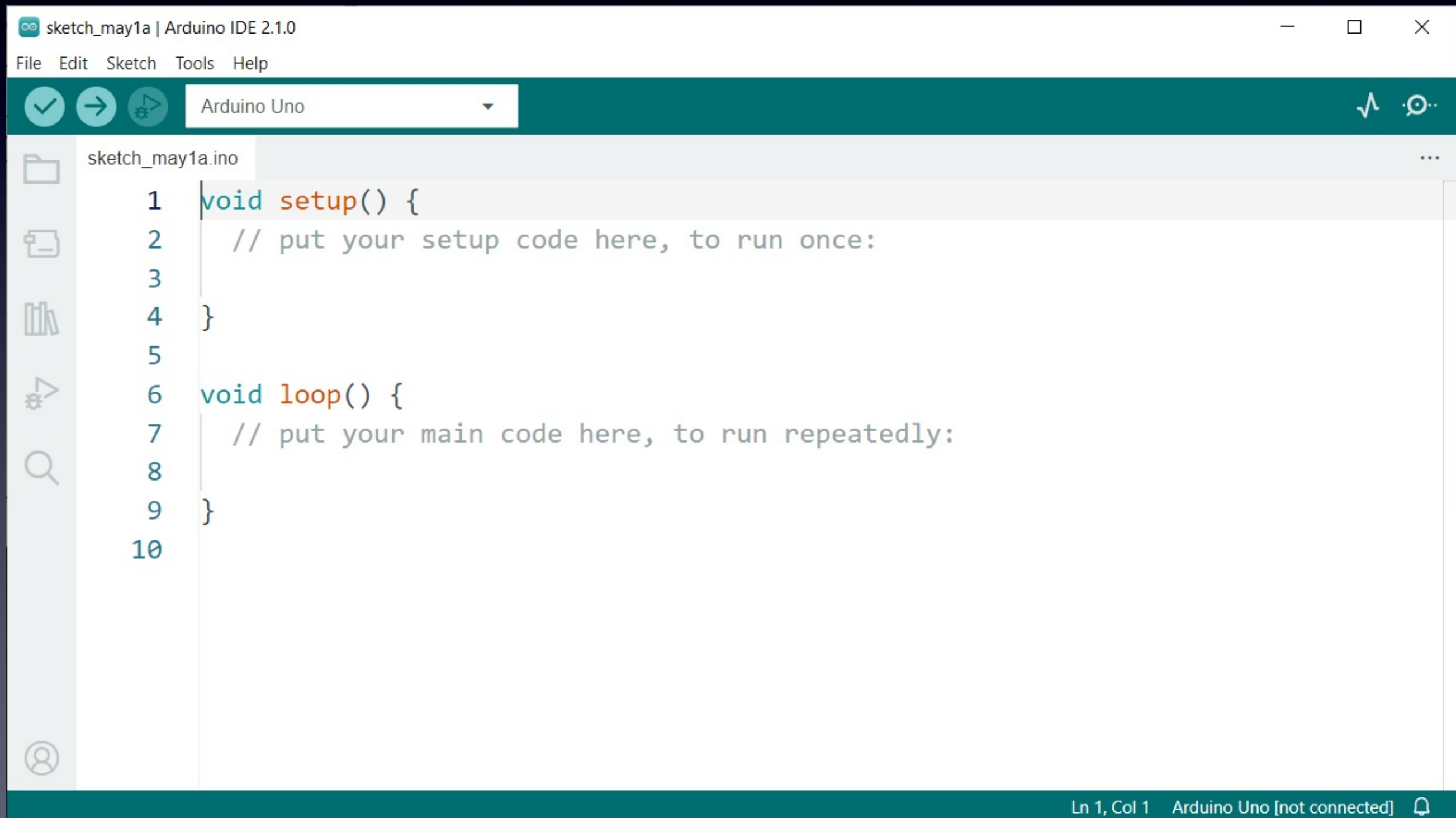
This shows a few more details:



IMPORTANT:
Make sure the
battery pack on
your ArduTouch
is *OFF*

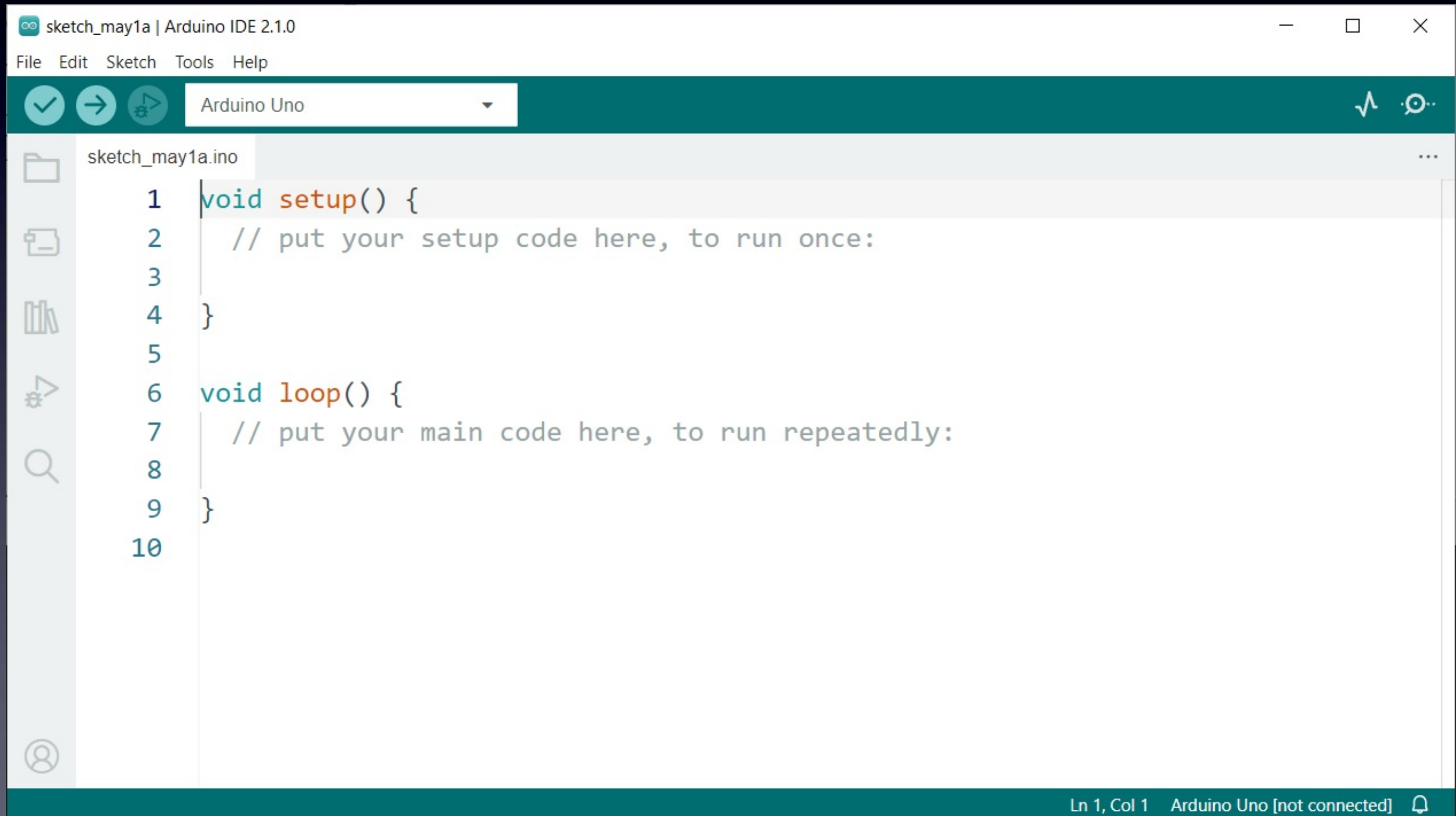
Arduino

After you download and install the Arduino software start it, and you will see a screen that looks like this:



Arduino

The **first time** you start your Arduino software you need to do **three things** to set things up:

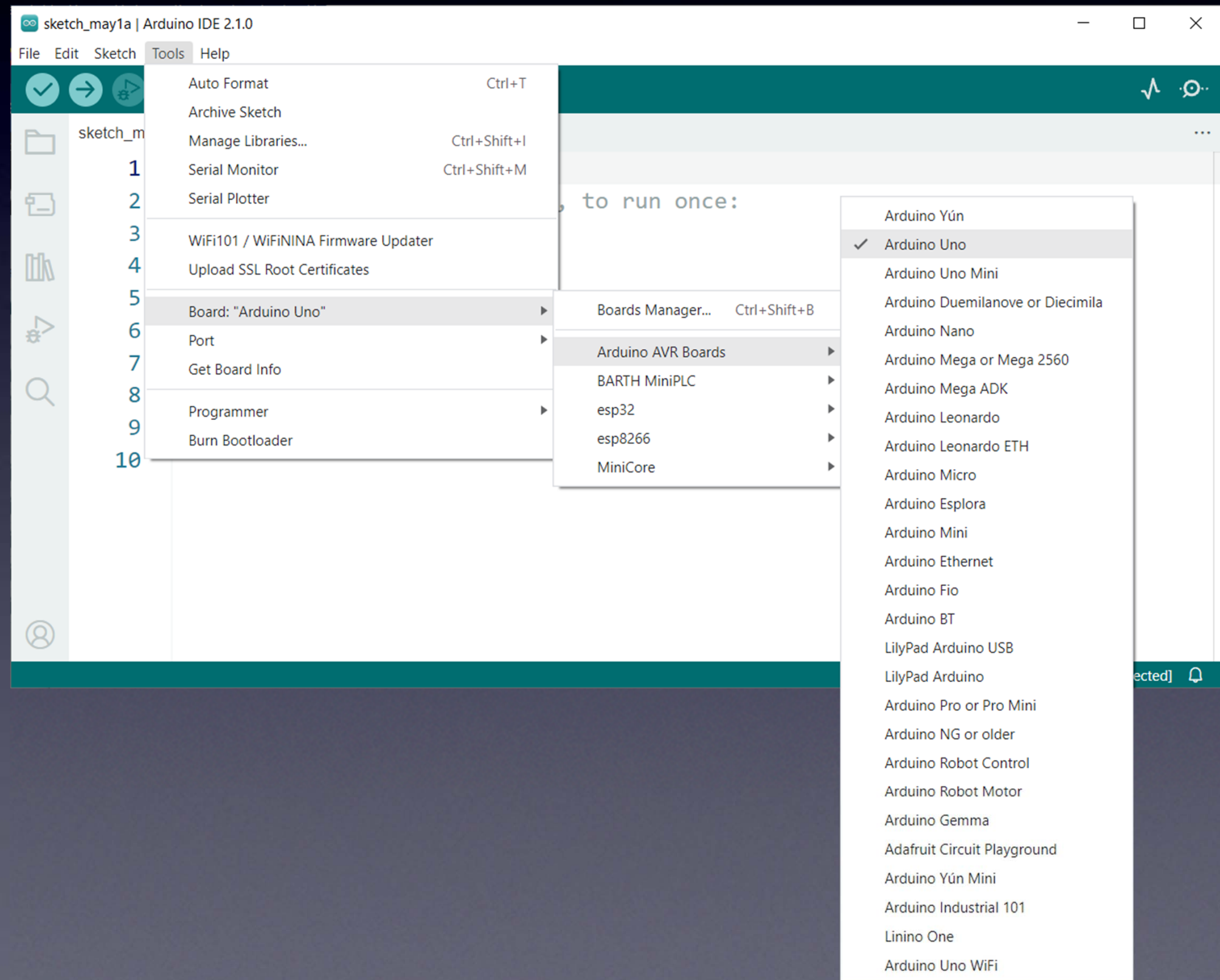


Arduino

The **first time** you start your Arduino software you need to do **three things** to set things up:

(1)
Choose “Uno”
as the Board

(Your
ArduTouch board
acts
just like
an
Arduino Uno board)

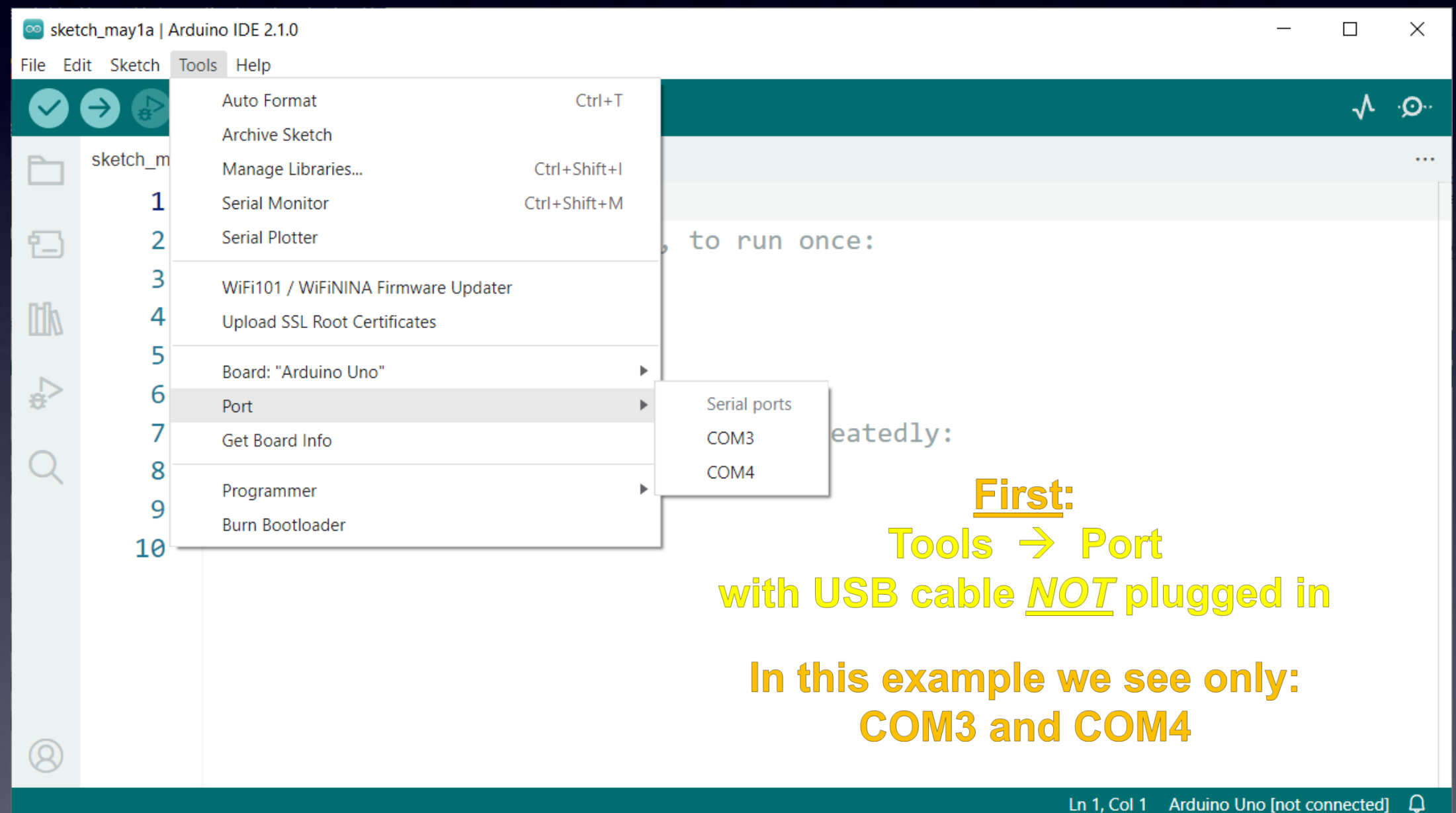


Arduino

The **first time** you start your Arduino software you need to do **three things** to set things up:

(2)

Choose the Port
(this will be different depending on your Operating System)



First:
Tools → Port
with USB cable NOT plugged in

In this example we see only:
COM3 and COM4

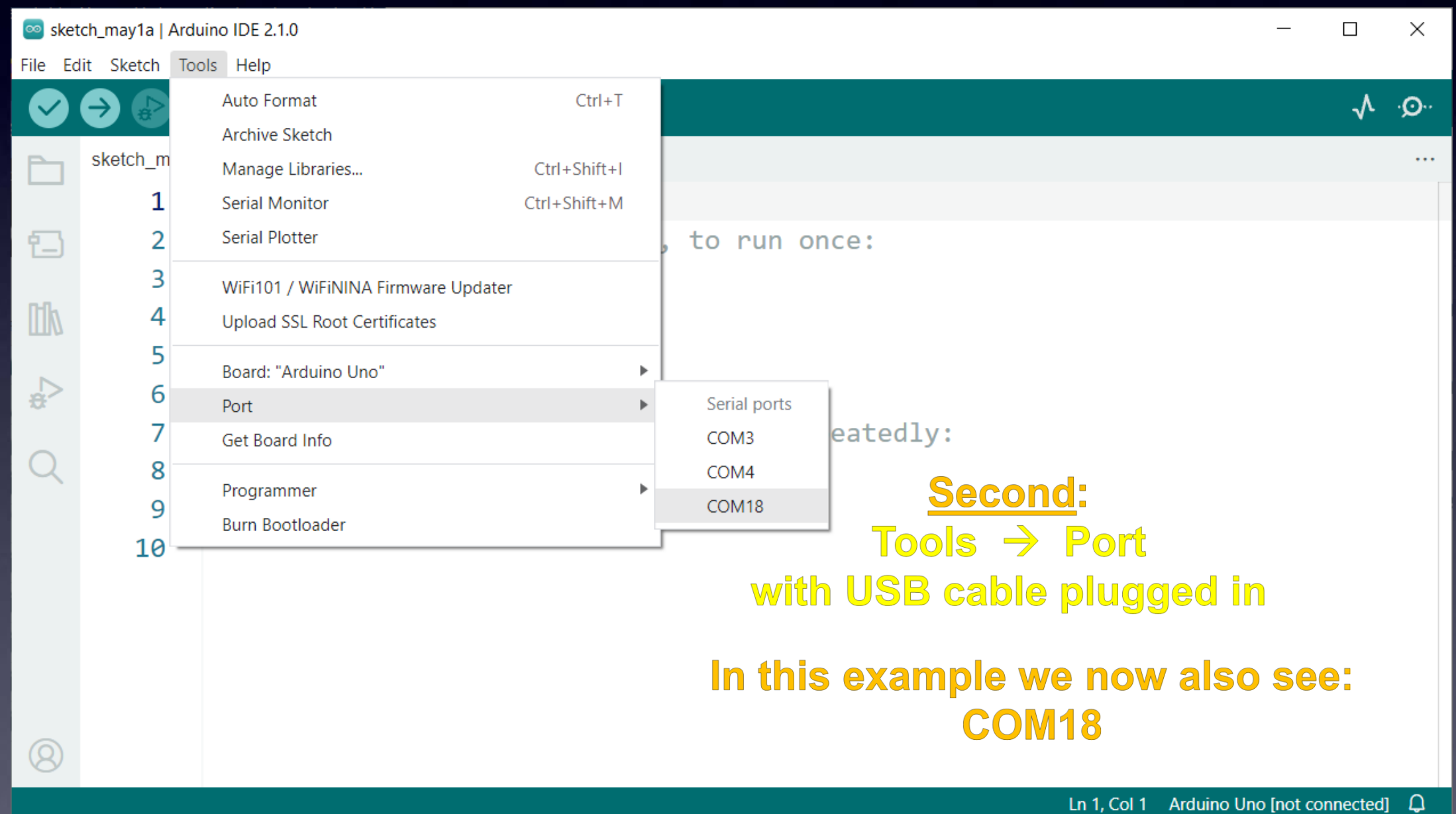
Arduino

The **first time** you start your Arduino software you need to do **three things** to set things up:

(2)

Choose the Port
(this will be different depending on your Operating System)

(After installing the driver for your USB-Serial cable, and plugging it in, your operating system will see a serial port and it appears here.)



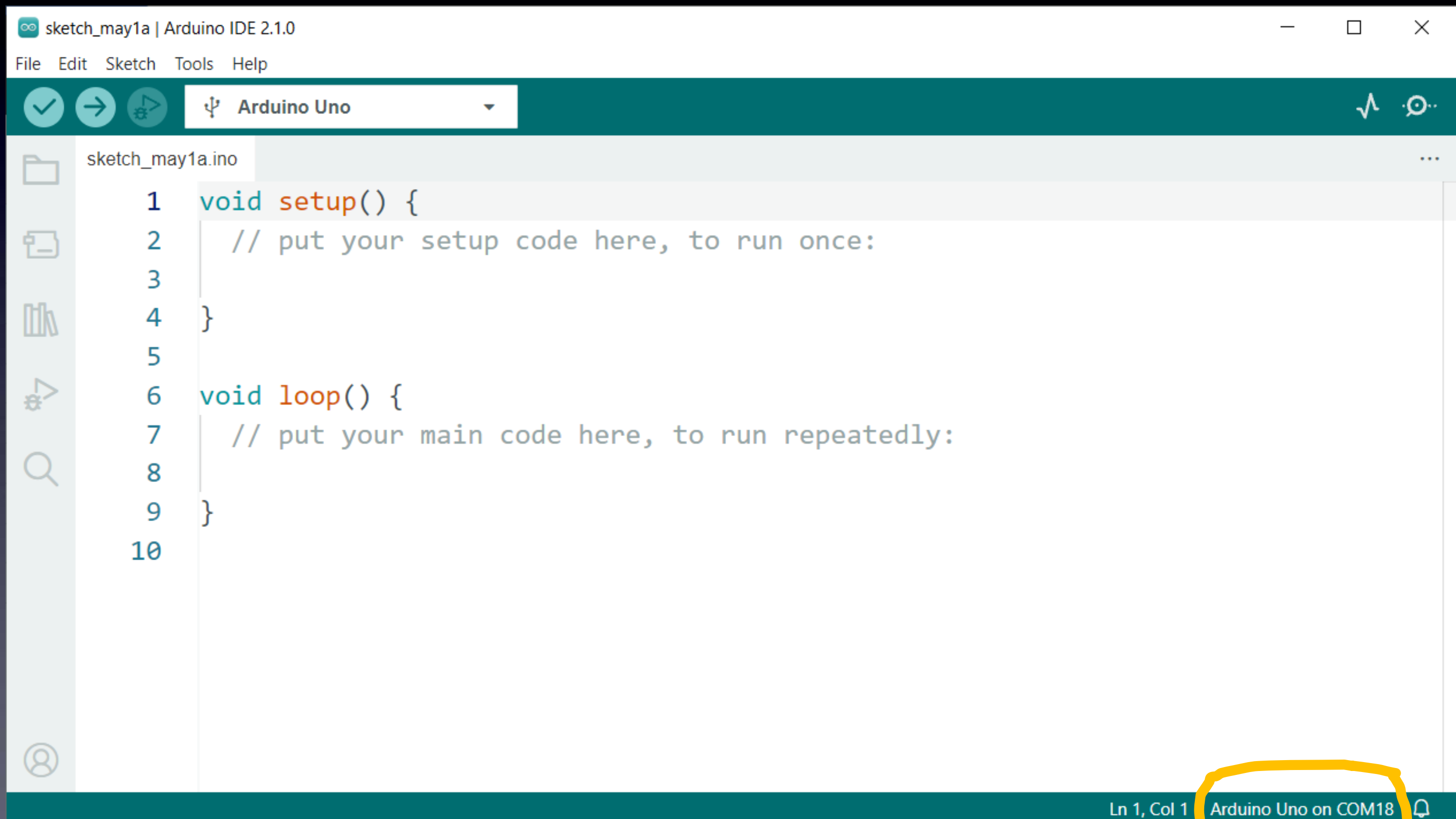
Second:
Tools → Port
with USB cable plugged in

In this example we now also see:
COM18

Choose the new port:
In this example: COM18

Arduino

Your Arduino software is almost ready...



Arduino

The *first time* you start your Arduino software you need to do *three things* to set things up:

(3)
Install
the
ArduTouch
library



Arduino

The *first time* you start your Arduino software you need to do *three things* to set things up:

(3)
Install
the
ArduTouch
library



Arduino

The *first time* you start your Arduino software you need to do *three things* to set things up:

(3)
Install
the
ArduTouch
library



Arduino

The *first time* you start your Arduino software you need to do *three things* to set things up:

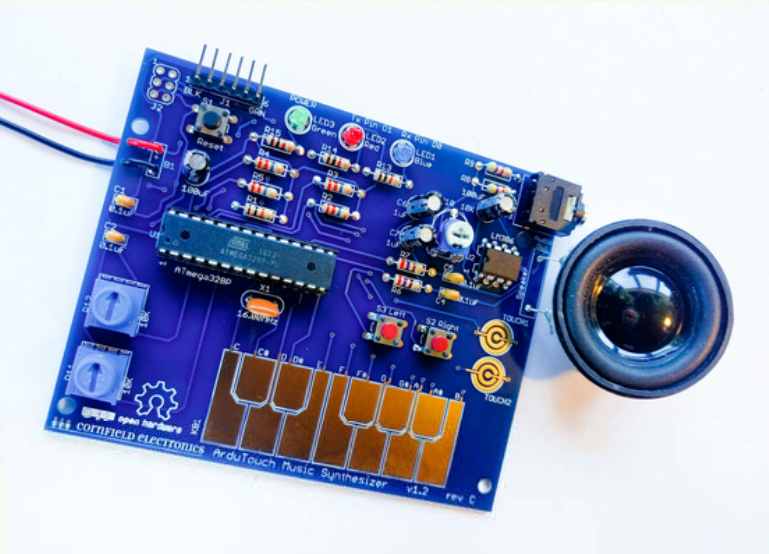
(3)
Install
the
ArduTouch
library

File Edit View History Bookmarks Tools Help

Cornfield Electronics :: Projects

https://cornfieldelectronics.com/cfe/projects.php?PHPSESSID=d5d4714nuevrq25drkkoir1m3

Project: ArduTouch Arduino-compatible Music Synthesizer kit
-- make way cool sounds and music!



==> **BUY an ArduTouch music synthesizer kit!** <==

Solder your ArduTouch kit together, and it works! You can make way wonderful music, sound, and noise. Use the ArduTouch Library or hack the existing sketches to create your own cool synthesizers. The documentation is getting good enough to learn how to use Digital Signal Processing (DSP) to make your own sounds for your own projects. (More documentation coming.)

This kit takes about 120 minutes to complete.

For **assembly instructions**, please see:
[ArduTouch assembly instructions for Rev C board](#)

older versions (before 2017):
[\(assembly instructions for Rev B, Rev A, and mono\)](#)

To program your ArduTouch music synthesizer kit, you'll need a USB-Serial TTL cable, such as an FTDI Friend or FTDI Cable, available all over the place. You can [purchase a nice one](#) from Cornfield Electronics. These USB-Serial TTL cables (made by Samurai Circuits), require a driver (from Silicon Labs):
Samurai Circuits board (SiLabs CP210x USB-to-Serial TTL) drivers:
[The latest drivers from SiLabs' website \(Windows, MacOS, Linux\)](#)

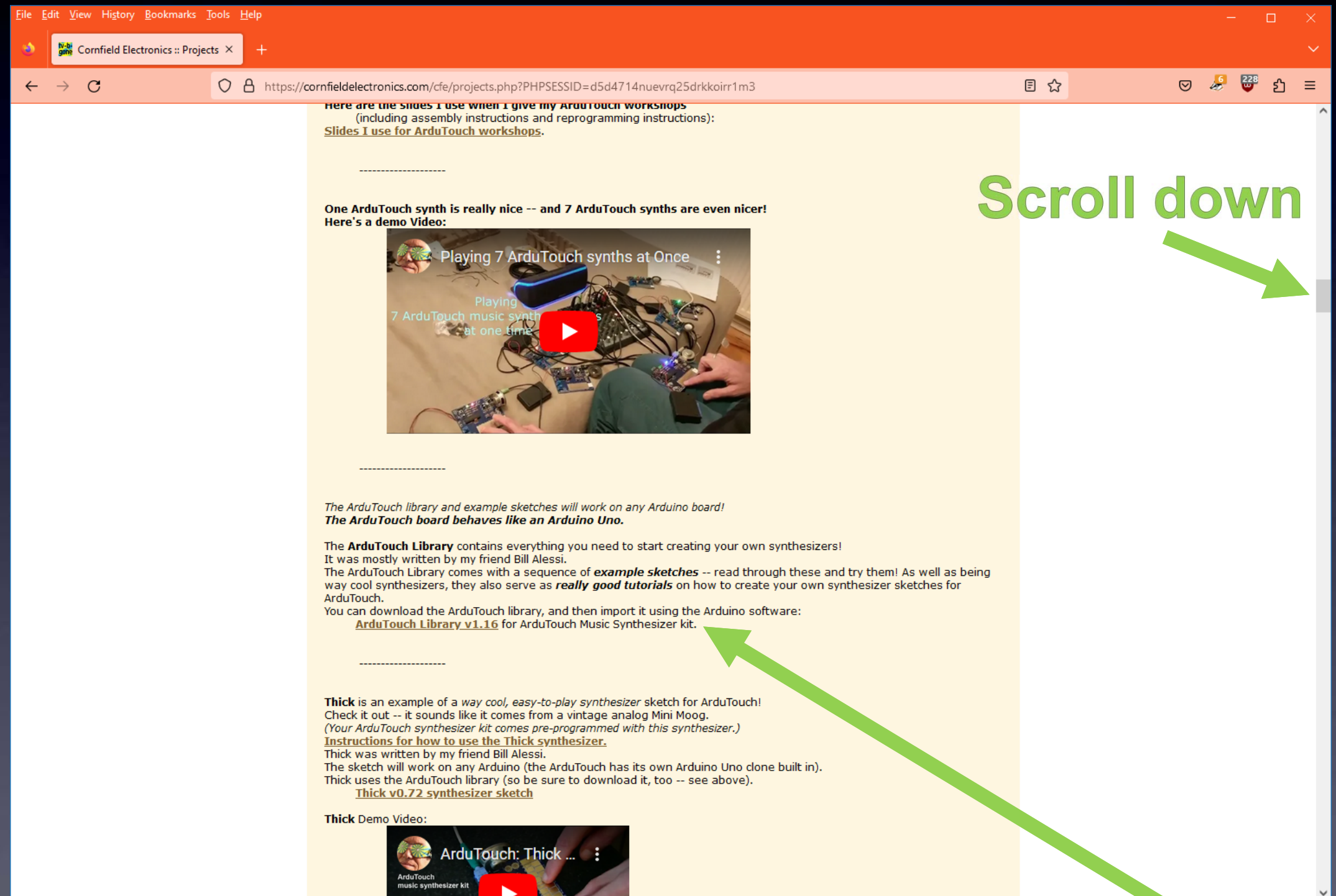
You will also need to download the free, open-source, Arduino software (for Windows, MacOS, or Linux).
[Arduino software](#) (the latest version is fine to use).

Scroll down

Arduino

The *first time* you start your Arduino software you need to do *three things* to set things up:

(3)
Install
the
ArduTouch
library

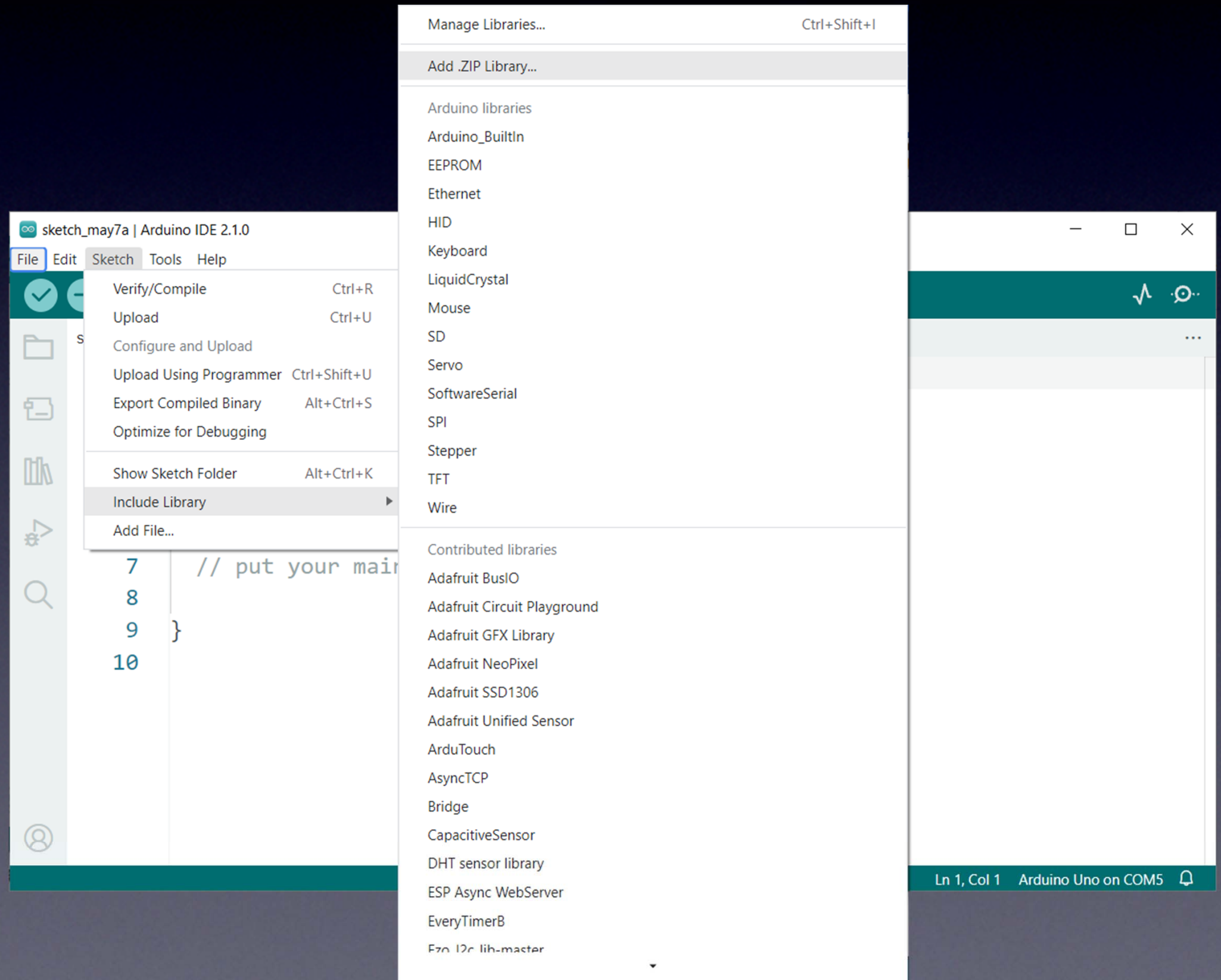


click this link to download the ArduTouch library

Arduino

The **first time** you start your Arduino software you need to do **three things** to set things up:

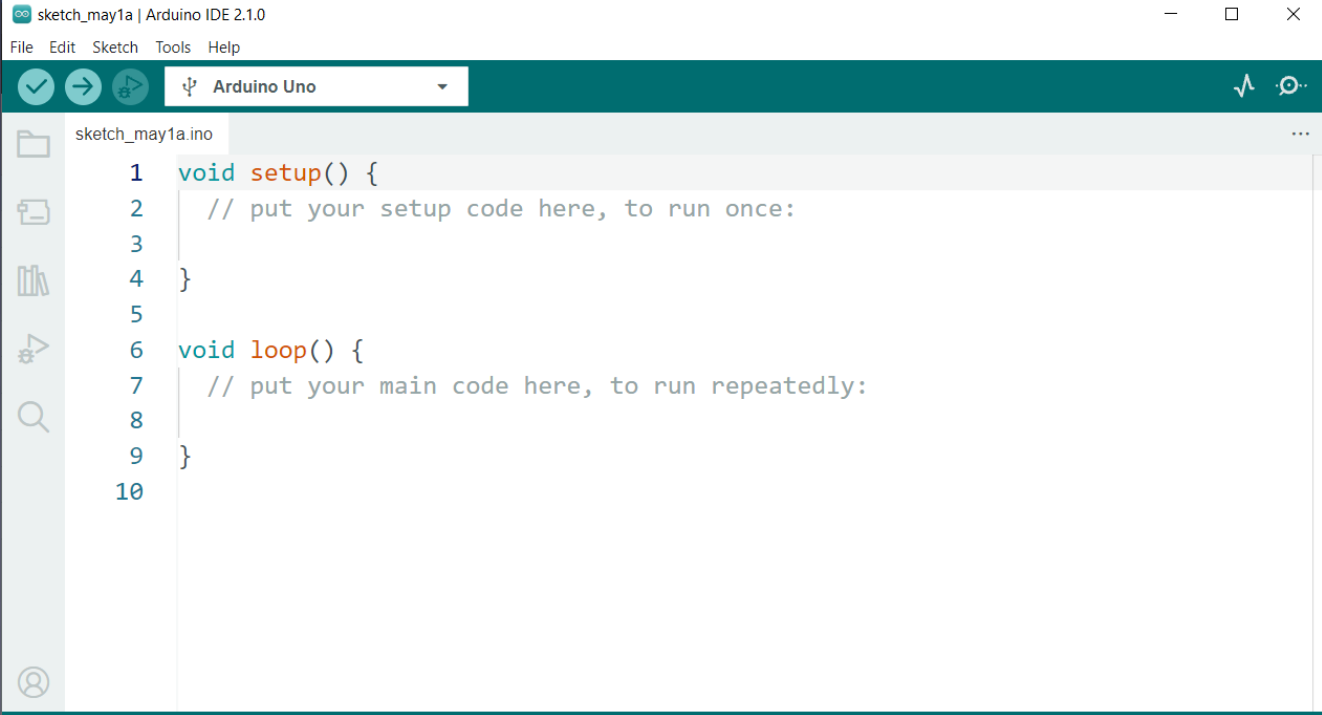
(3)
Install
the
ArduTouch
library



Arduino

Your Arduino software is now ready

to program a new synth sketch
into your ArduTouch !

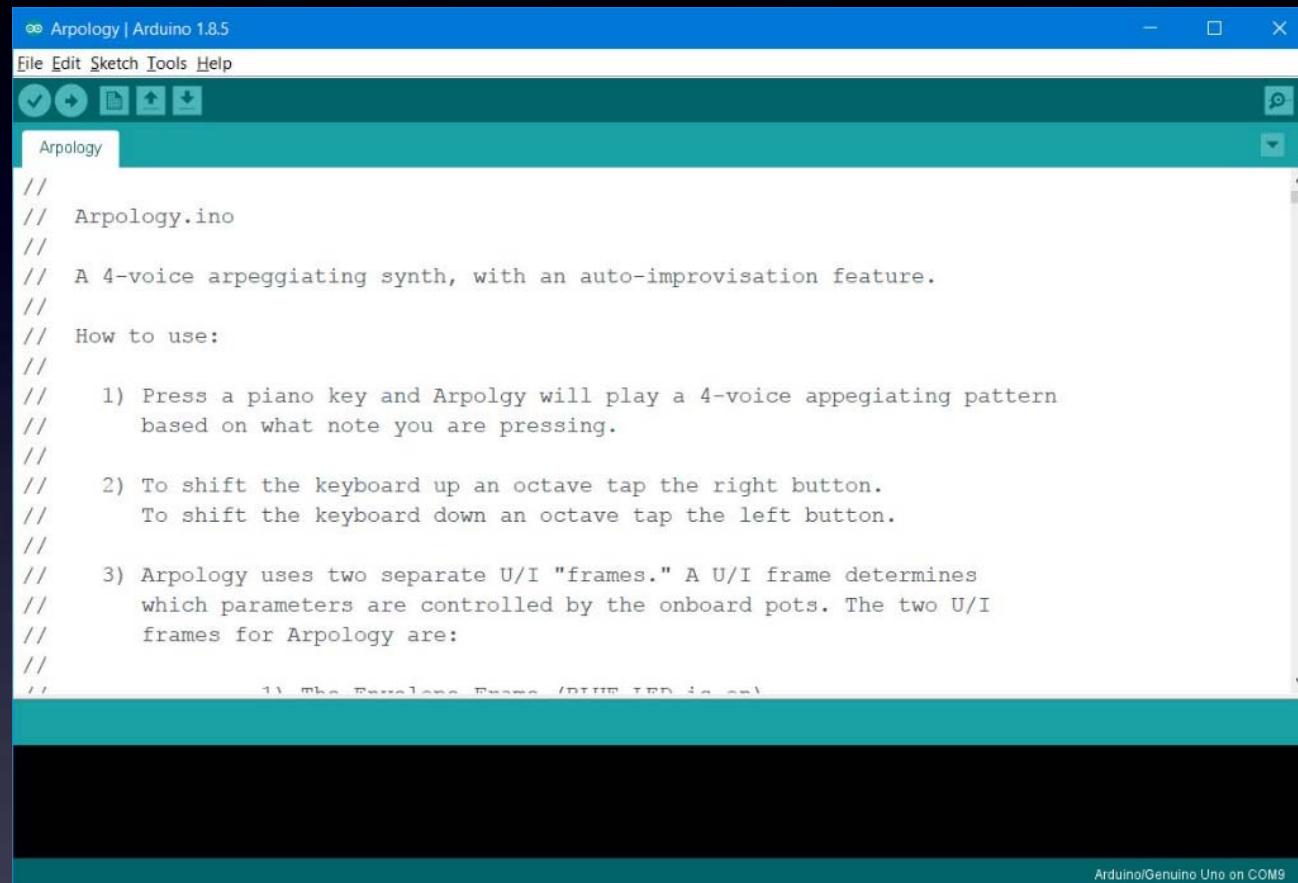


```
sketch_may1a | Arduino IDE 2.1.0
File Edit Sketch Tools Help
[Icons] Arduino Uno
sketch_may1a.ino
1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
10
Ln 1, Col 1 Arduino Uno on COM18
```

Arduino

Designed for non-geeky artists

Download
a new
ArduTouch
synth “sketch”

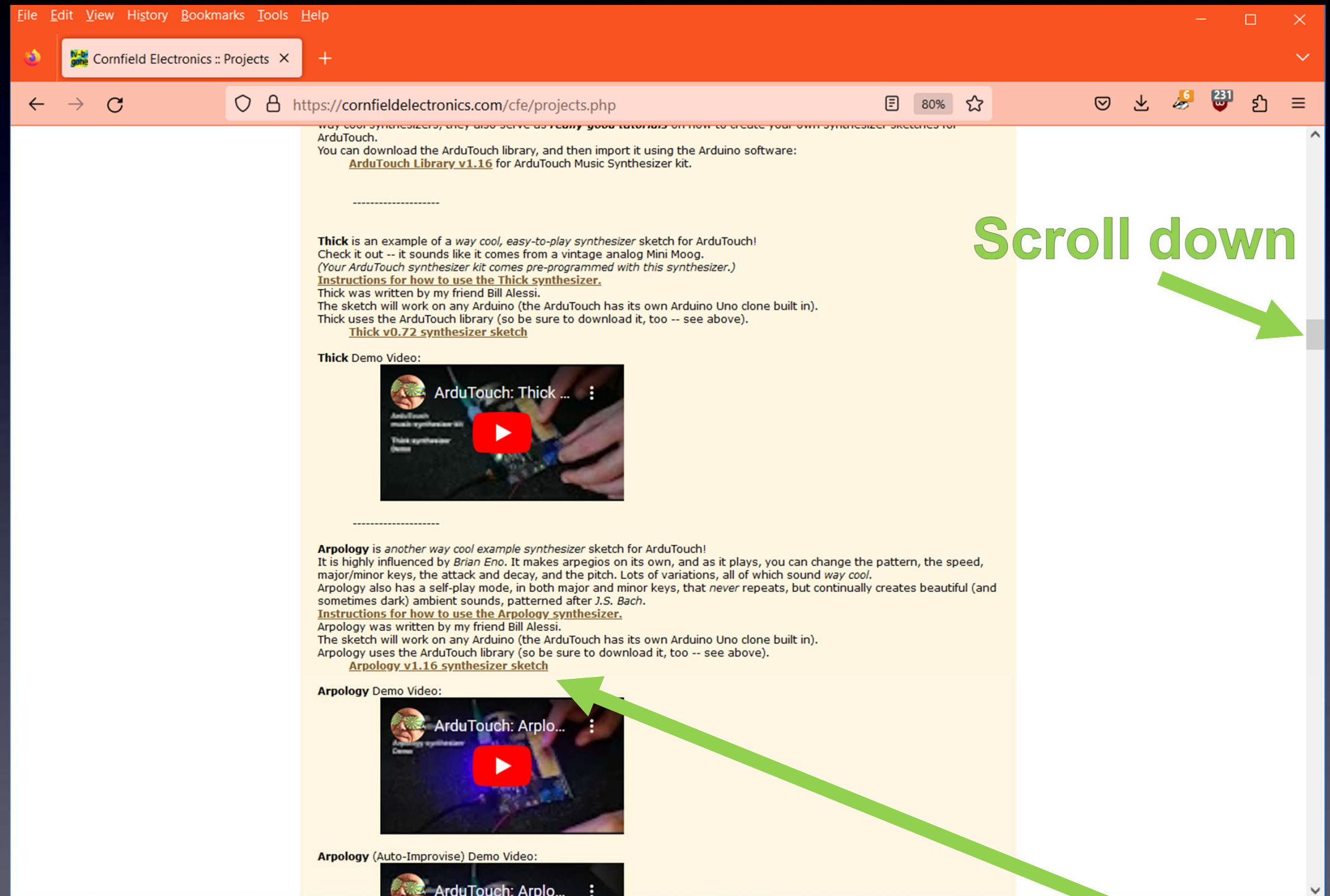
A screenshot of the Arduino IDE interface. The title bar reads "Arpology | Arduino 1.8.5". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for opening, saving, and running. The main text area shows the beginning of a C++ sketch named "Arpology.ino". The code includes comments describing it as a 4-voice arpeggiating synth with an auto-improvisation feature. It provides instructions on how to use the device, such as pressing piano keys to trigger patterns and using buttons to shift octaves. The status bar at the bottom indicates "Arduino/Genuino Uno on COM9".

```
//  
//  Arpology.ino  
//  
//  A 4-voice arpeggiating synth, with an auto-improvisation feature.  
//  
//  How to use:  
//  
//    1) Press a piano key and Arpology will play a 4-voice arpeggiating pattern  
//       based on what note you are pressing.  
//  
//    2) To shift the keyboard up an octave tap the right button.  
//       To shift the keyboard down an octave tap the left button.  
//  
//    3) Arpology uses two separate U/I "frames." A U/I frame determines  
//       which parameters are controlled by the onboard pots. The two U/I  
//       frames for Arpology are:  
//  
//    1) The Envelope Frame (BLUE LED is on)
```

“Sketch” :
an Arduino program

Arduino

Download a new ArduTouch synth “sketch”



Scroll down

click link to download a synth “sketch”

Arduino

Download a new ArduTouch synth “sketch”

File Edit View History Bookmarks Tools Help

Cornfield Electronics :: Projects X +

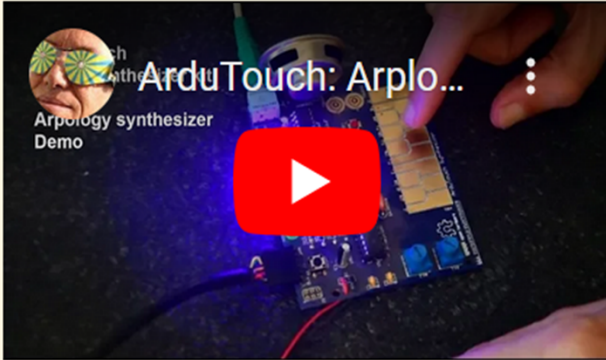
← → ↻ 🔒 https://cornfieldelectronics.com/cfe/projects.php 120% ☆

Arpology is another way cool example synthesizer sketch for ArduTouch!
It is highly influenced by *Brian Eno*. It makes arpeggios on its own, and as it plays, you can change the pattern, the speed, major/minor keys, the attack and decay, and the pitch. Lots of variations, all of which sound way cool.
Arpology also has a self-play mode, in both major and minor keys, that *never* repeats, but continually creates beautiful (and sometimes dark) ambient sounds, patterned after *J.S. Bach*.

Instructions for how to use the Arpology synthesizer.
Arpology was written by my friend Bill Alessi.
The sketch will work on any Arduino (the ArduTouch has its own Arduino Uno clone built in).
Arpology uses the ArduTouch library (so be sure to download it, too -- see above).

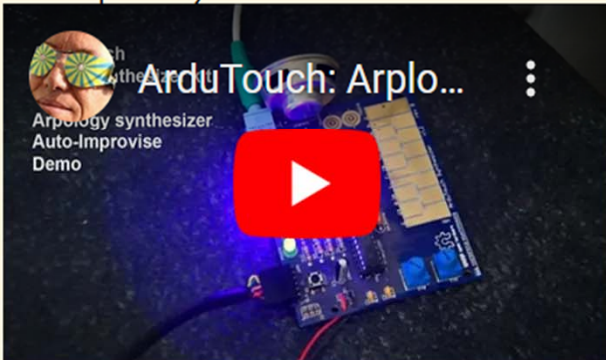
Arpology v1.16 synthesizer sketch

Arpology Demo Video:



ArduTouch: Arplo...
Arpology synthesizer Demo

Arpology (Auto-Improvise) Demo Video:



ArduTouch: Arplo...
Arpology synthesizer Auto-Improvise Demo

Check it out!

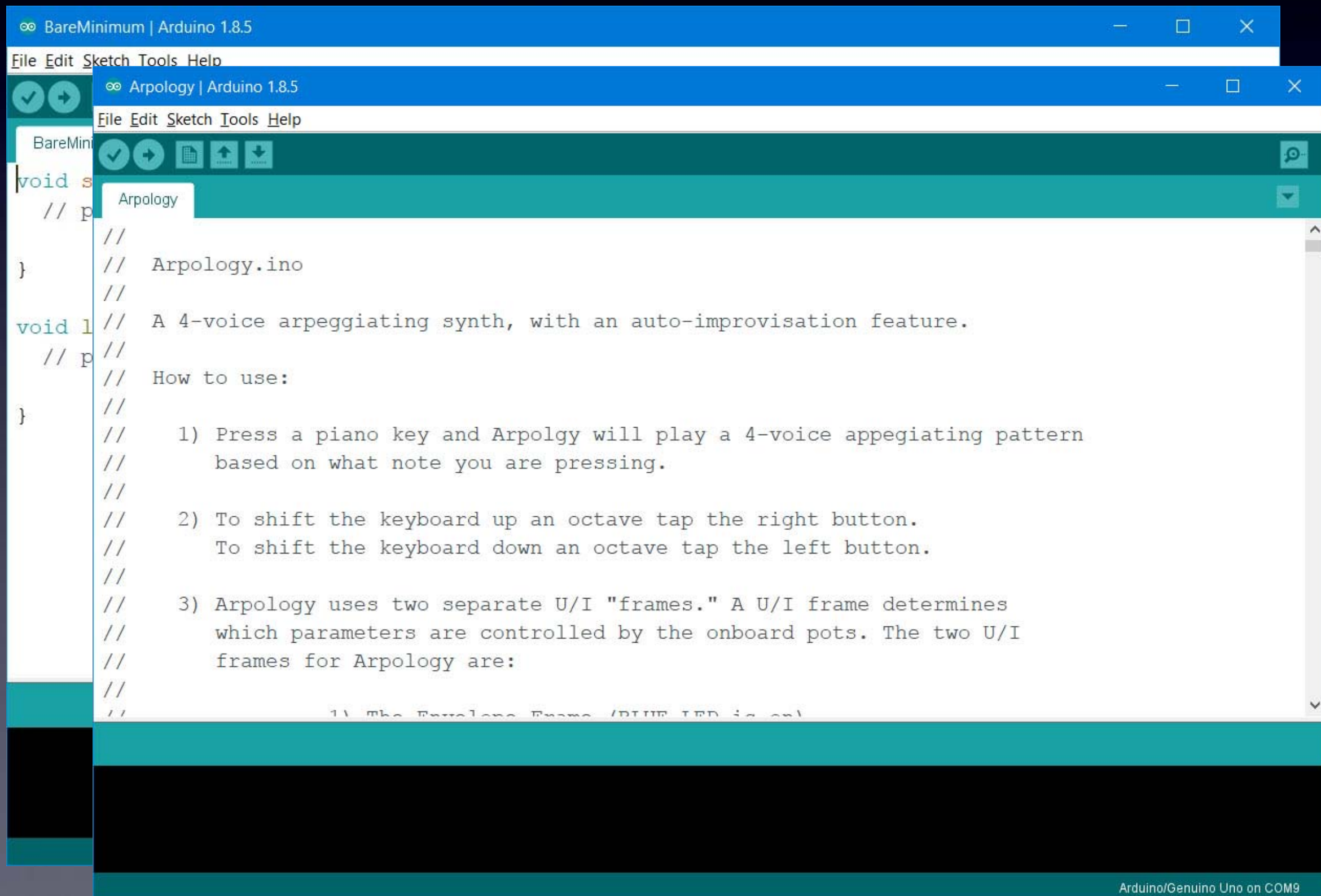
Also
available
for each synth:

- Demo Videos
- Instructions

Arduino

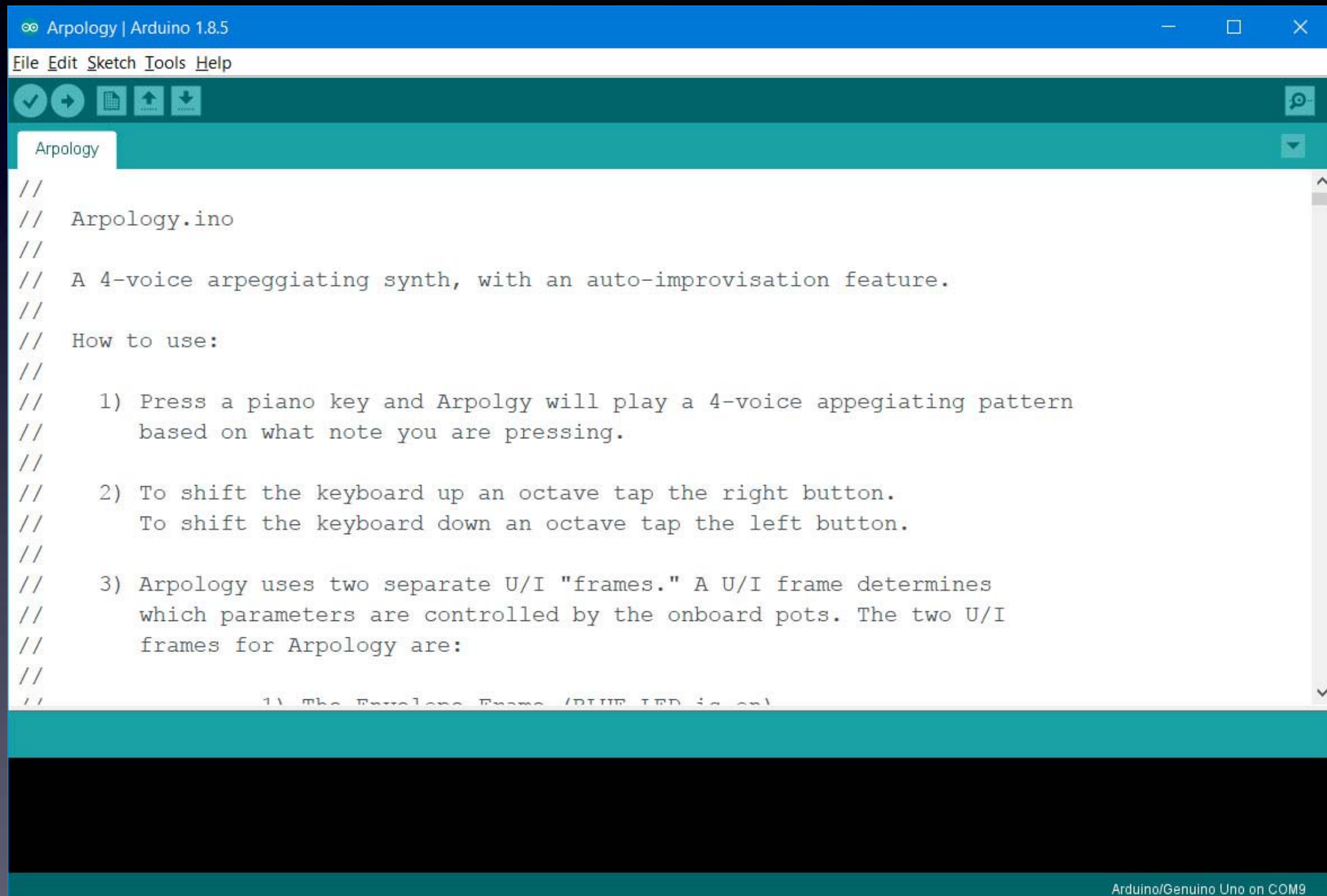
You can open the ArduTouch synth sketch:
File → Open...

(I opened “Arpology here)



Arduino

You can now program your ArduTouch with a new synth sketch !

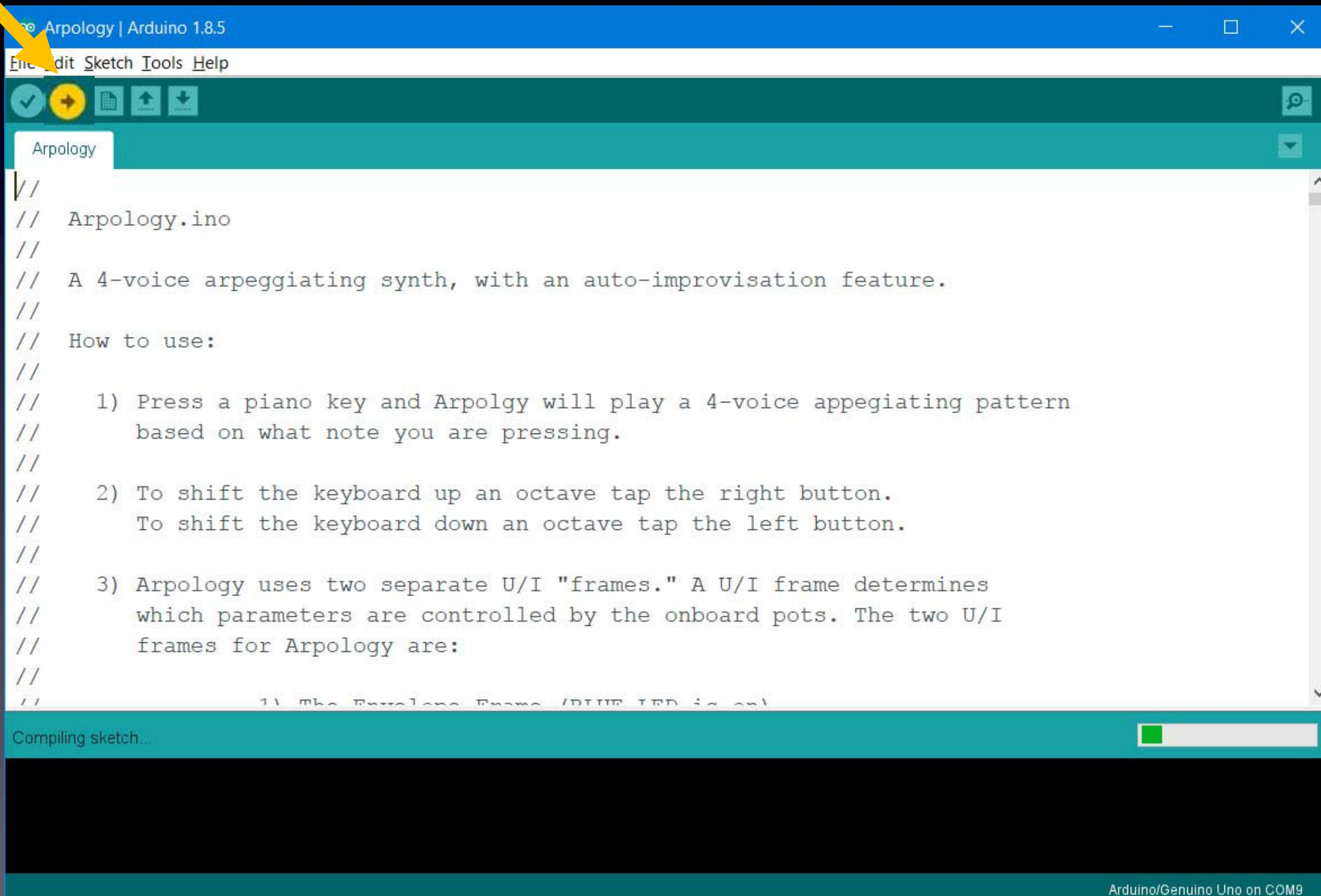
A screenshot of the Arduino IDE interface. The title bar reads "Arpology | Arduino 1.8.5". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checking, running, saving, and uploading. A tab labeled "Arpology" is active. The main text area contains the following code:

```
//  
//  Arpology.ino  
//  
//  A 4-voice arpeggiating synth, with an auto-improvisation feature.  
//  
//  How to use:  
//  
//    1) Press a piano key and Arpology will play a 4-voice arpeggiating pattern  
//       based on what note you are pressing.  
//  
//    2) To shift the keyboard up an octave tap the right button.  
//       To shift the keyboard down an octave tap the left button.  
//  
//    3) Arpology uses two separate U/I "frames." A U/I frame determines  
//       which parameters are controlled by the onboard pots. The two U/I  
//       frames for Arpology are:  
//  
//    1) The Envelope Frame (BLUE LED is on)
```

The status bar at the bottom right indicates "Arduino/Genuino Uno on COM9".

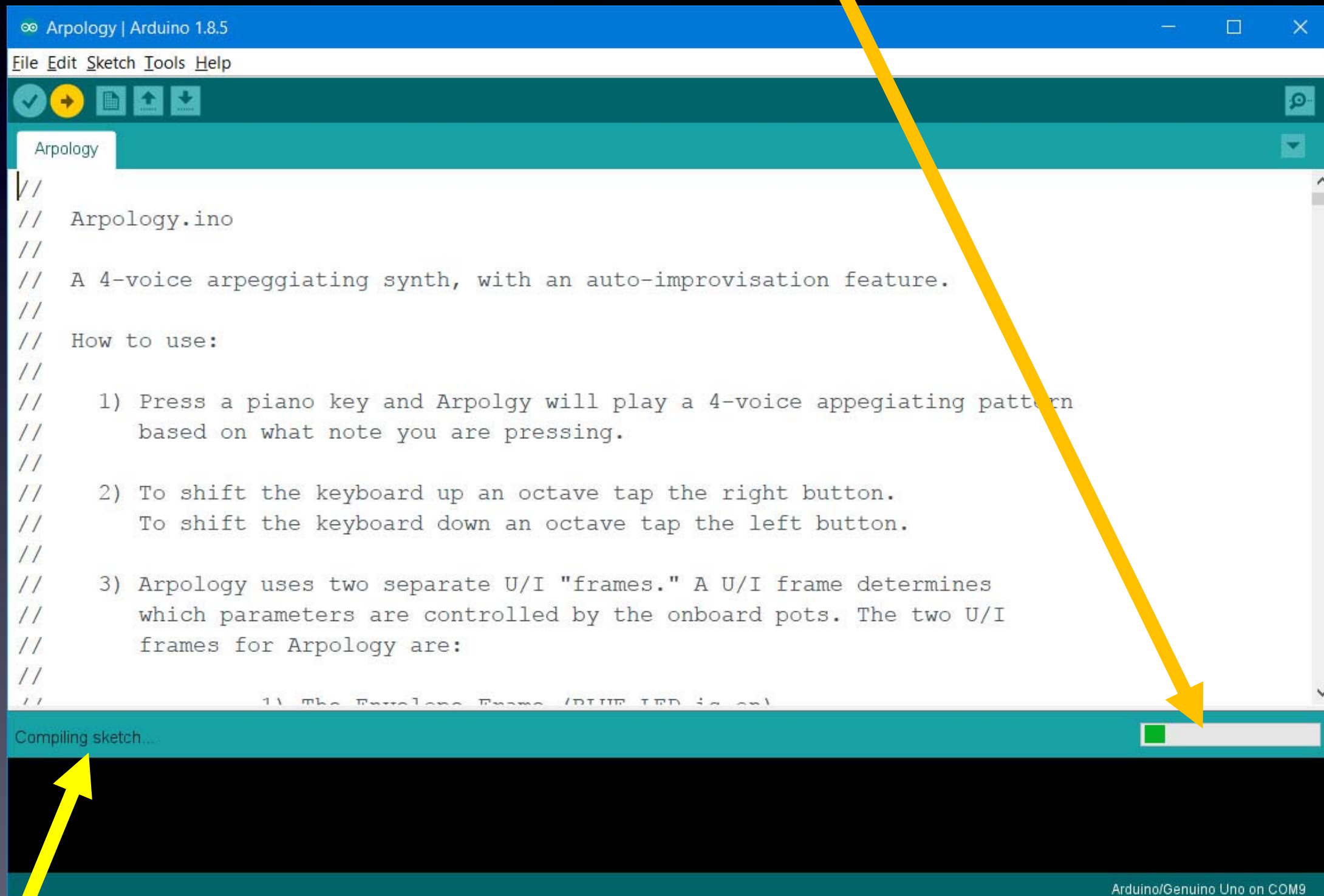
Arduino

With the USB-Serial cable connected to your ArduTouch board press the Upload button



Arduino

While uploading, you will see a progress bar...



...and when it's completed successfully, it says: "Upload done"

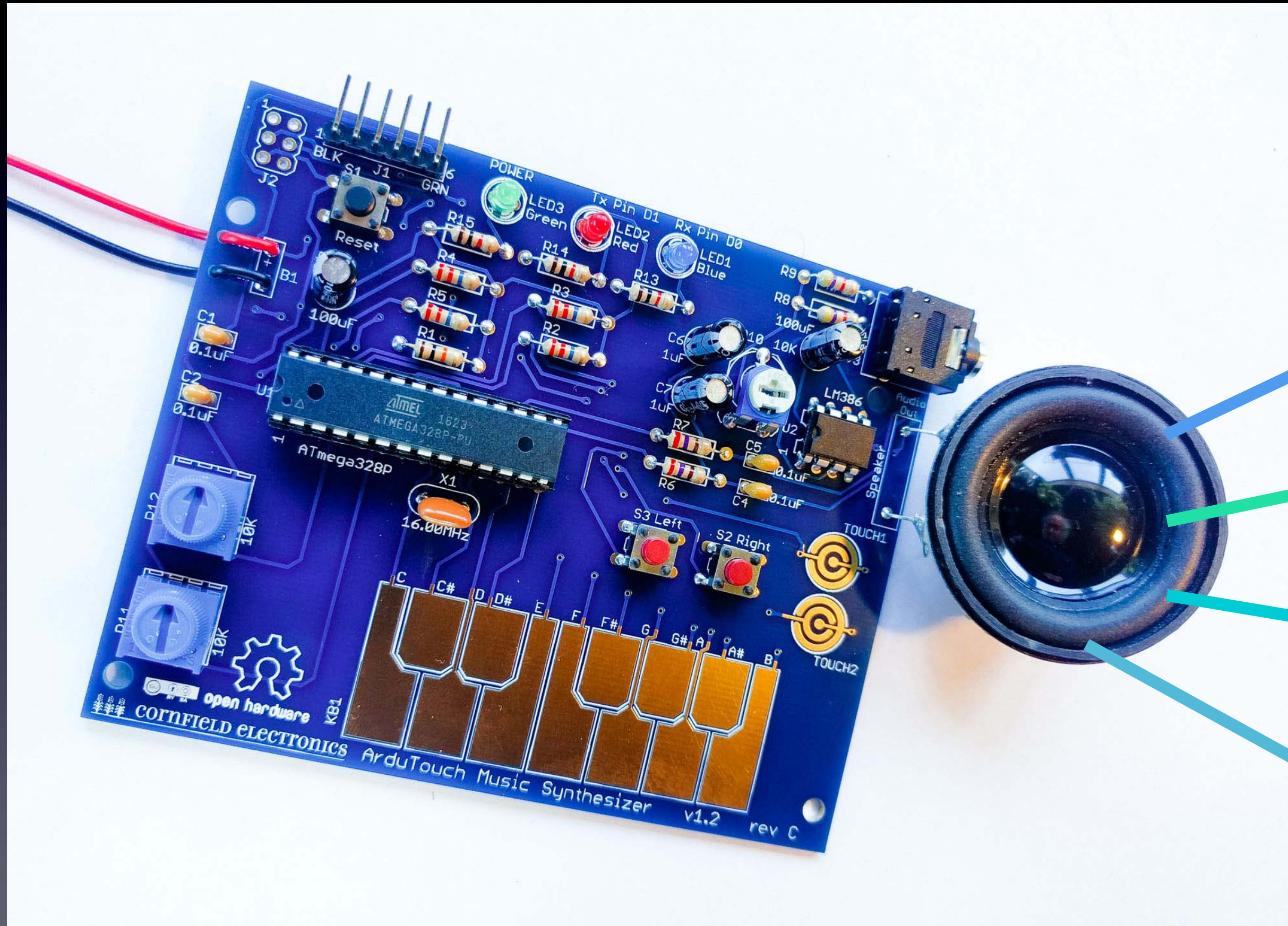
ArduTouch

**Disconnect your ArduTouch board
from the USB-Serial cable,**

turn on your battery pack,

And...

Let's make new noise!



Please Remember:

to

Wash your hands

after soldering

Learn to Solder

with ArduTouch Music Synthesizer kit

and make music, sound, and noise!

Mitch Altman

Chief Scientist, **Cornfield Electronics**, San Francisco, CA

Inventor of **TV-B-Gone** universal remote controls

Co-founder of **3Ware** (successful Silicon Valley startup)

Pioneer of **VR** (in the mid-1980s)

Founding mentor at **HAX** (1st and biggest hardware accelerator)

Co-founder of **Noisebridge** (San Francisco hackerspace)

email: mitch@CornfieldElectronics.com

site: www.CornfieldElectronics.com

facebook: maltman23

flickr: maltman23

WeChat: mitchaltman

Fediverse: @maltman23@mastodon.social

Patreon: mitchaltman

fablab|nk



CORNFIELD ELECTRONICS