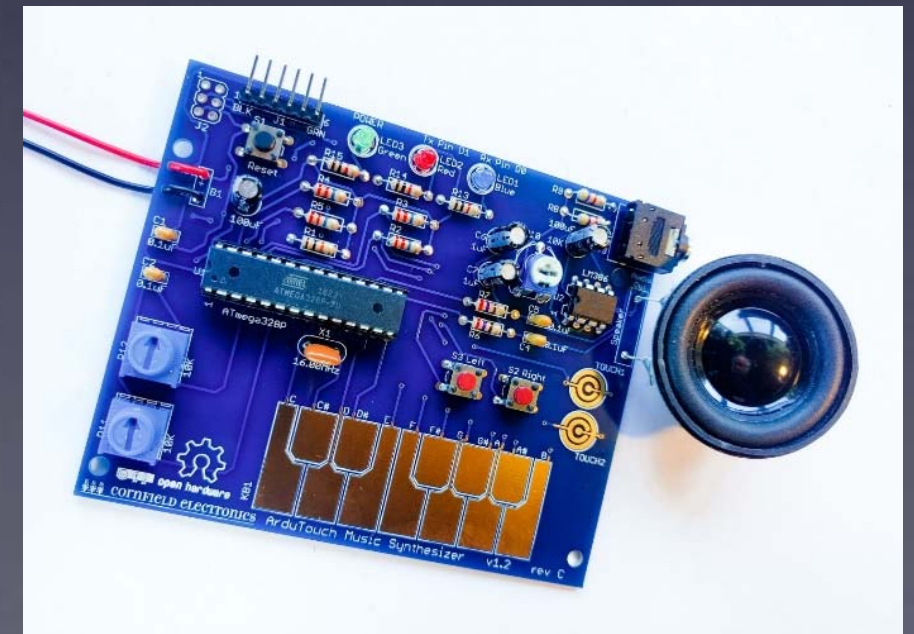




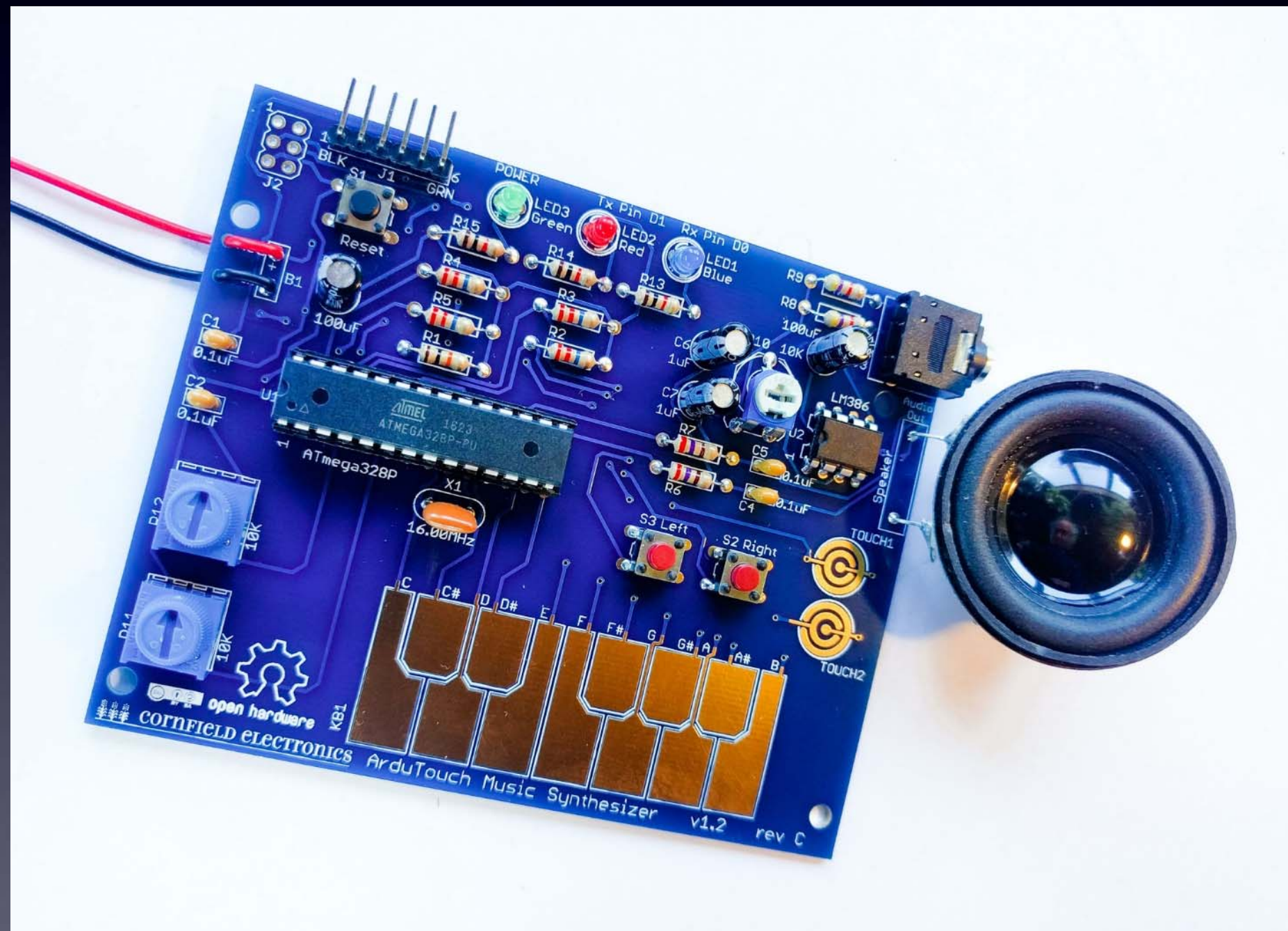
# Syllabus

- 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

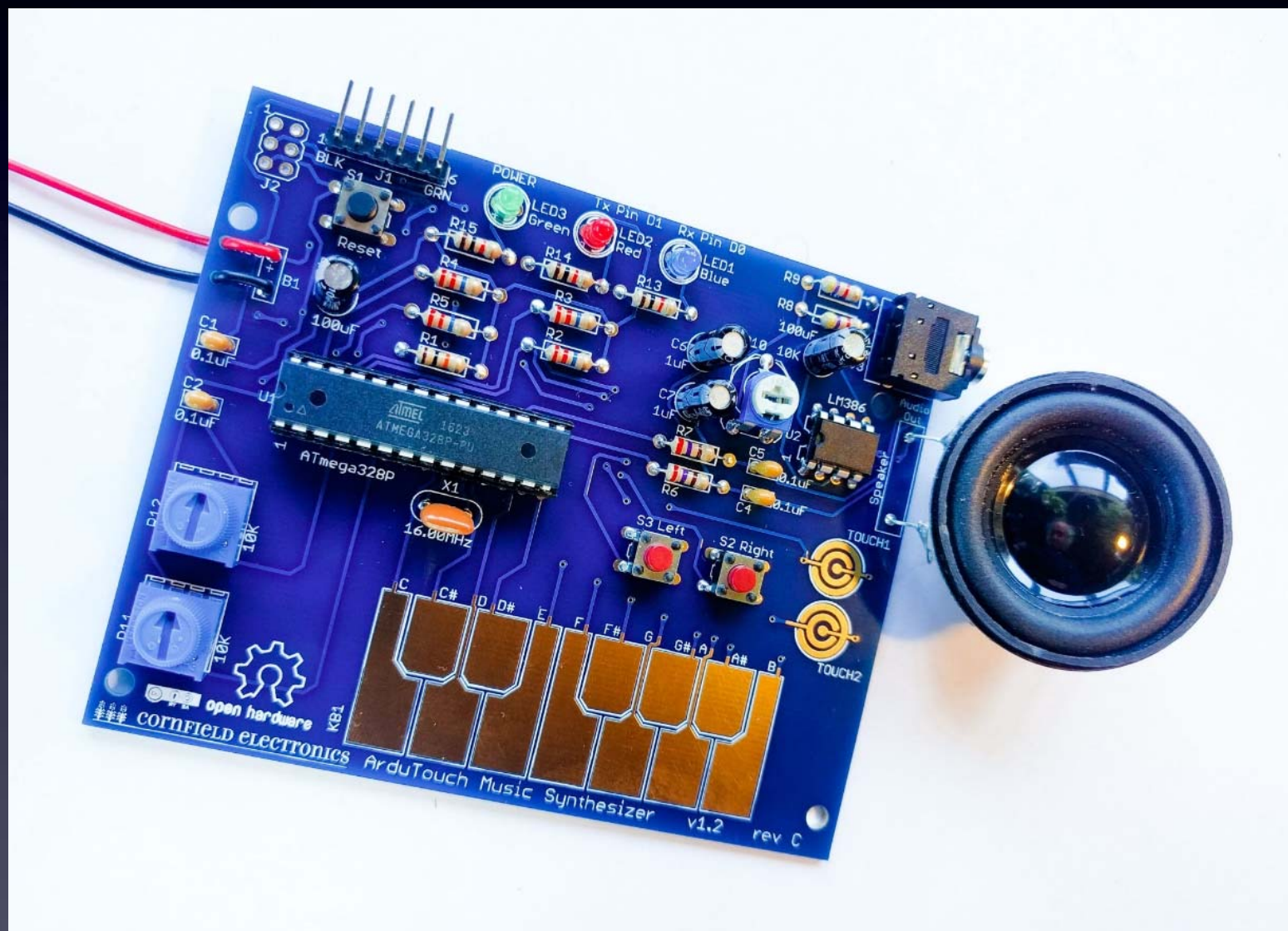


# ArduTouch Music Synthesizer

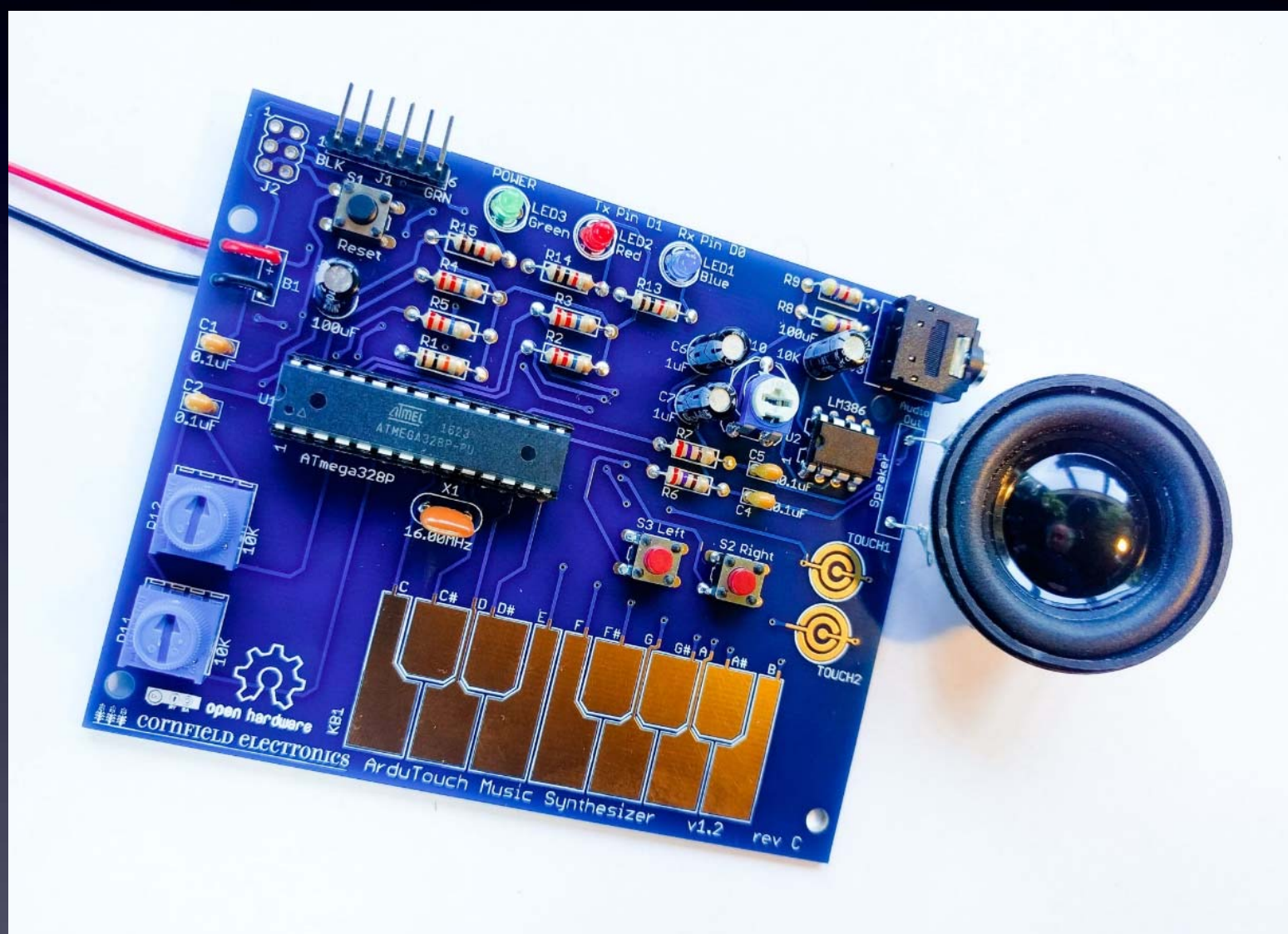


rev C

# ArduTouch

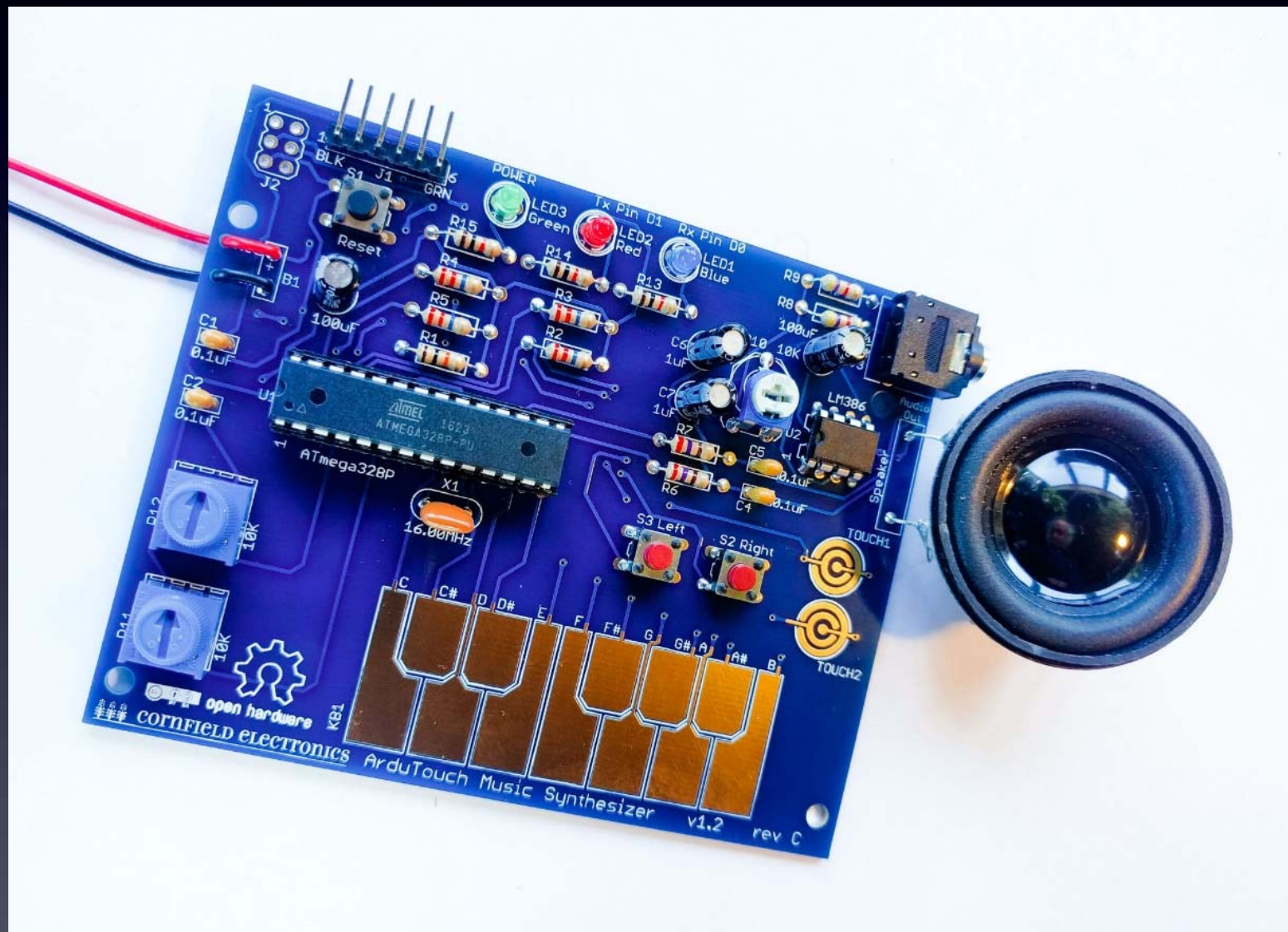


# ArduTouch



Great for  
learning  
to solder

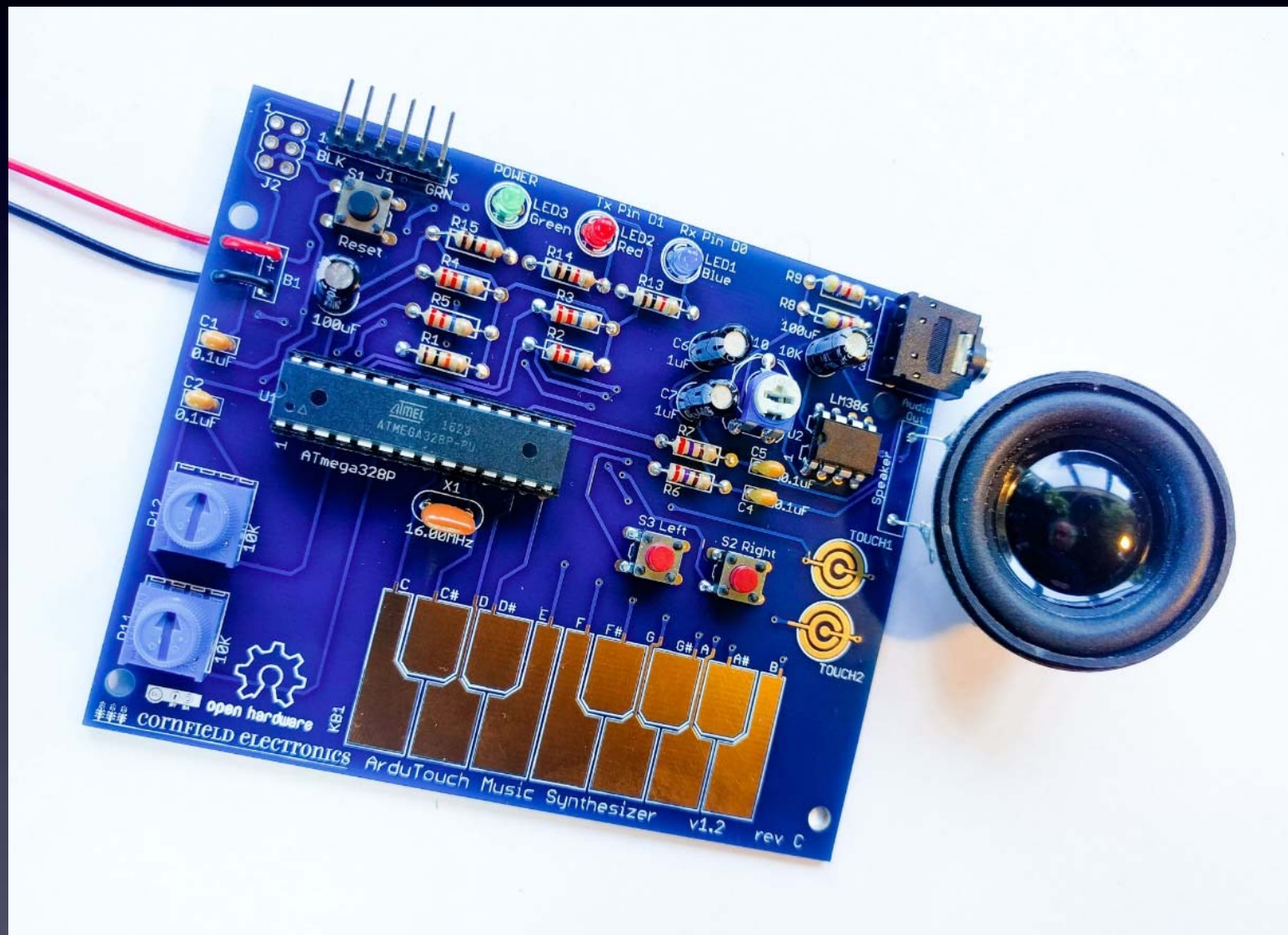
# ArduTouch



Solder it together  
– and it works!

And you can also  
*program*  
your own synthesizers

# ArduTouch



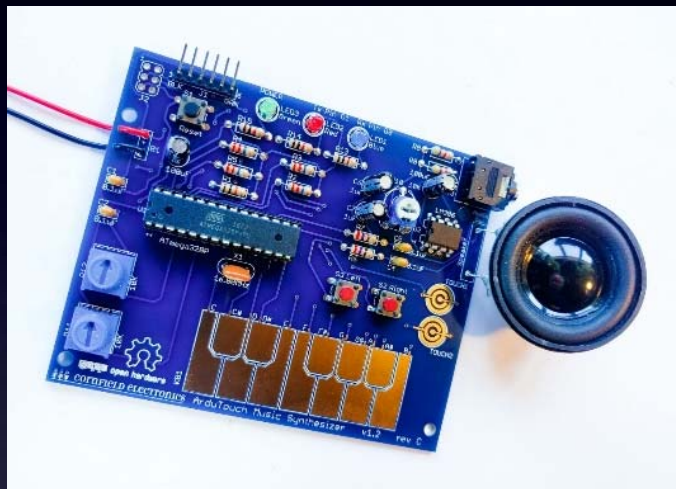
Solder it together  
– and it works!

And you can also  
*program*  
your own synthesizers

You can also  
*learn*  
Digital Signal Processing



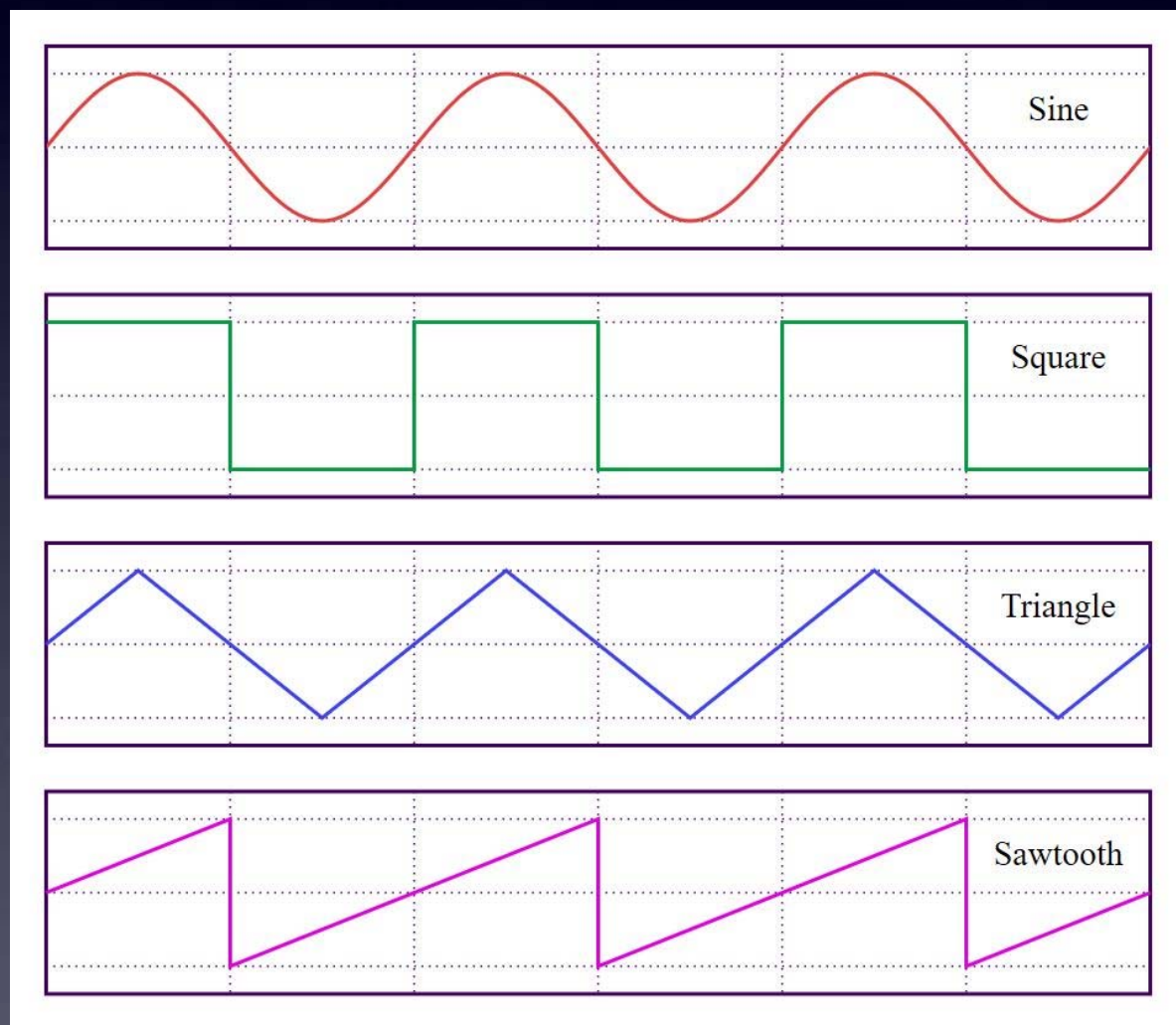
# ArduTouch



*Live demo*

# Some Types of Synthesizers

## Analog



### Modular Analog Synthesizer:

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

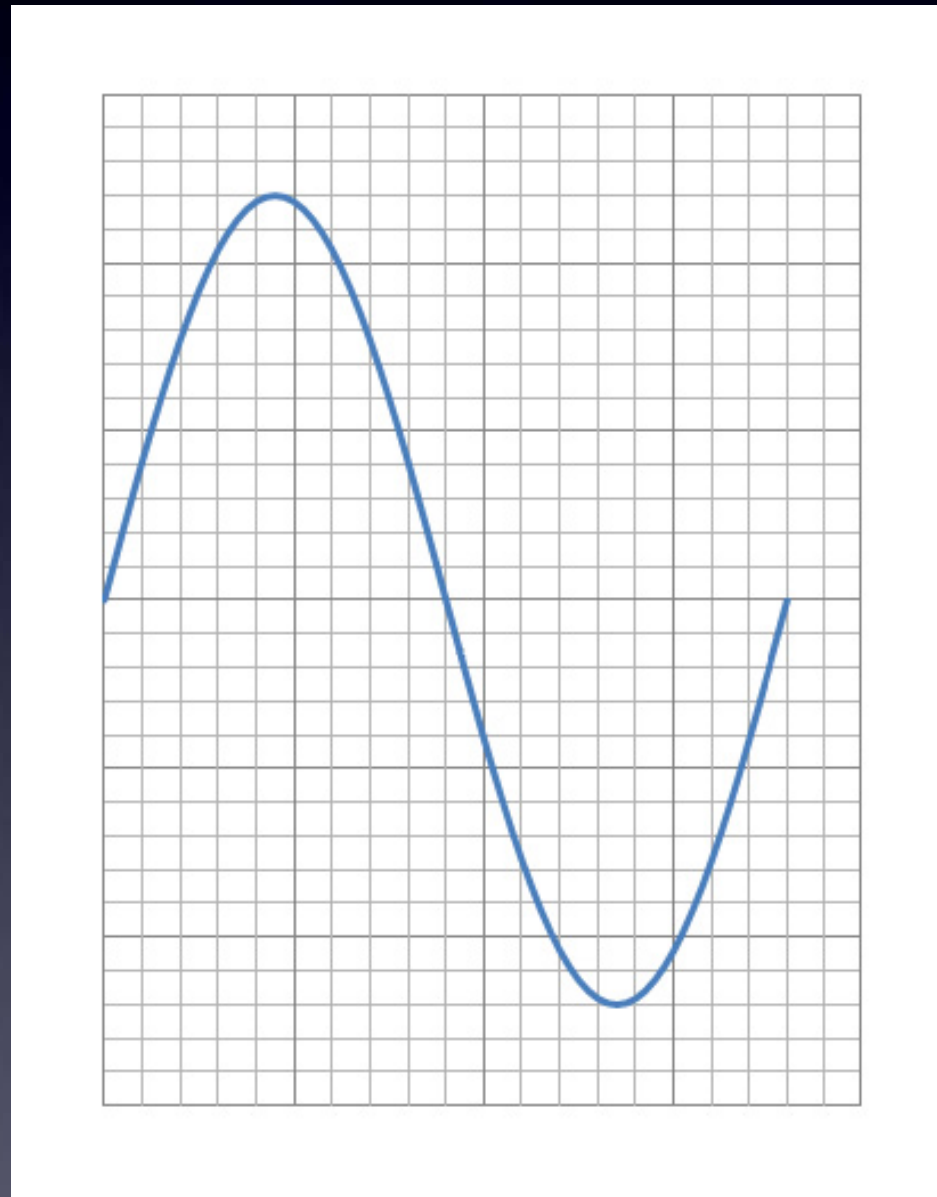
# Some Types of Synthesizers

## Digital

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

# Digital Signal Processing

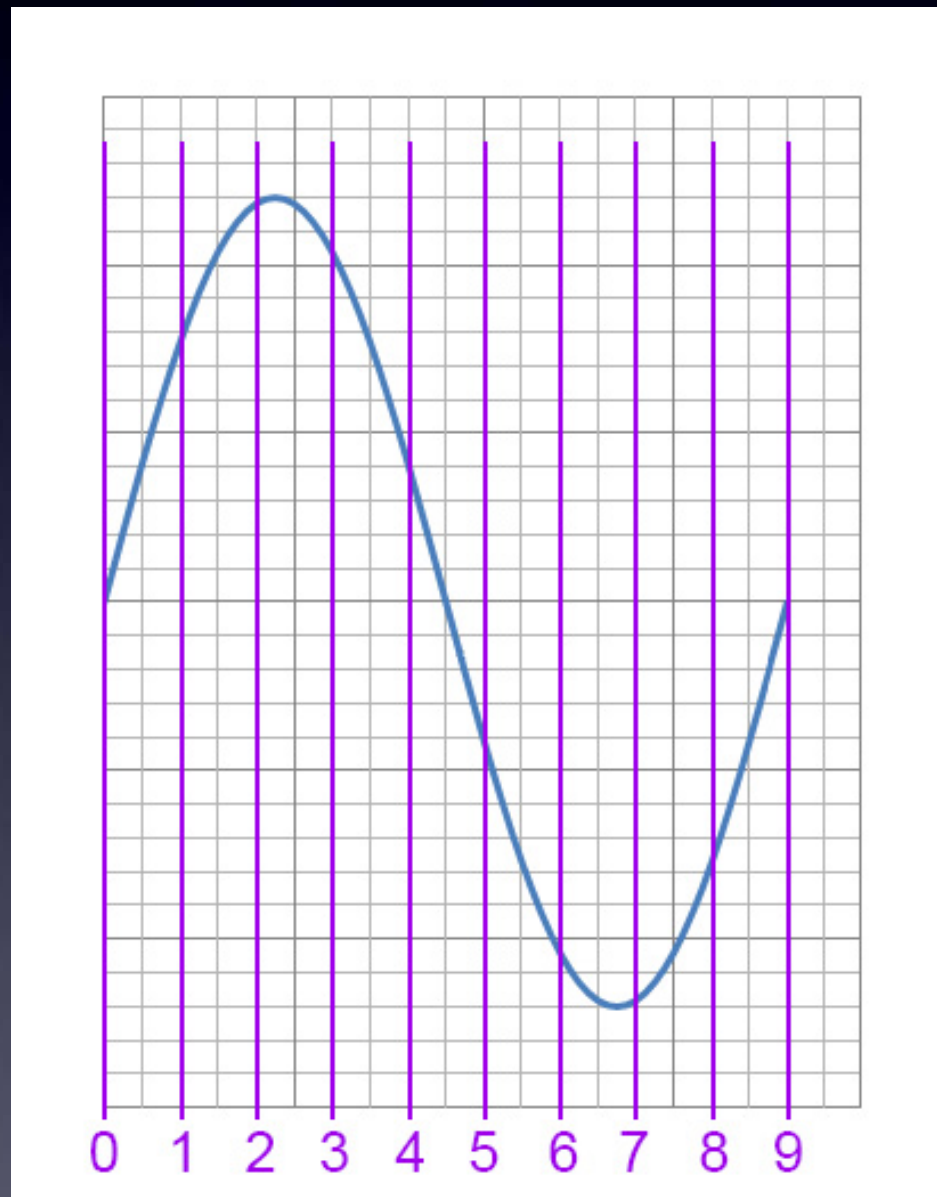
Analog waveform



# Digital Signal Processing

To record it digitally

First slice it  
(equal time slices)

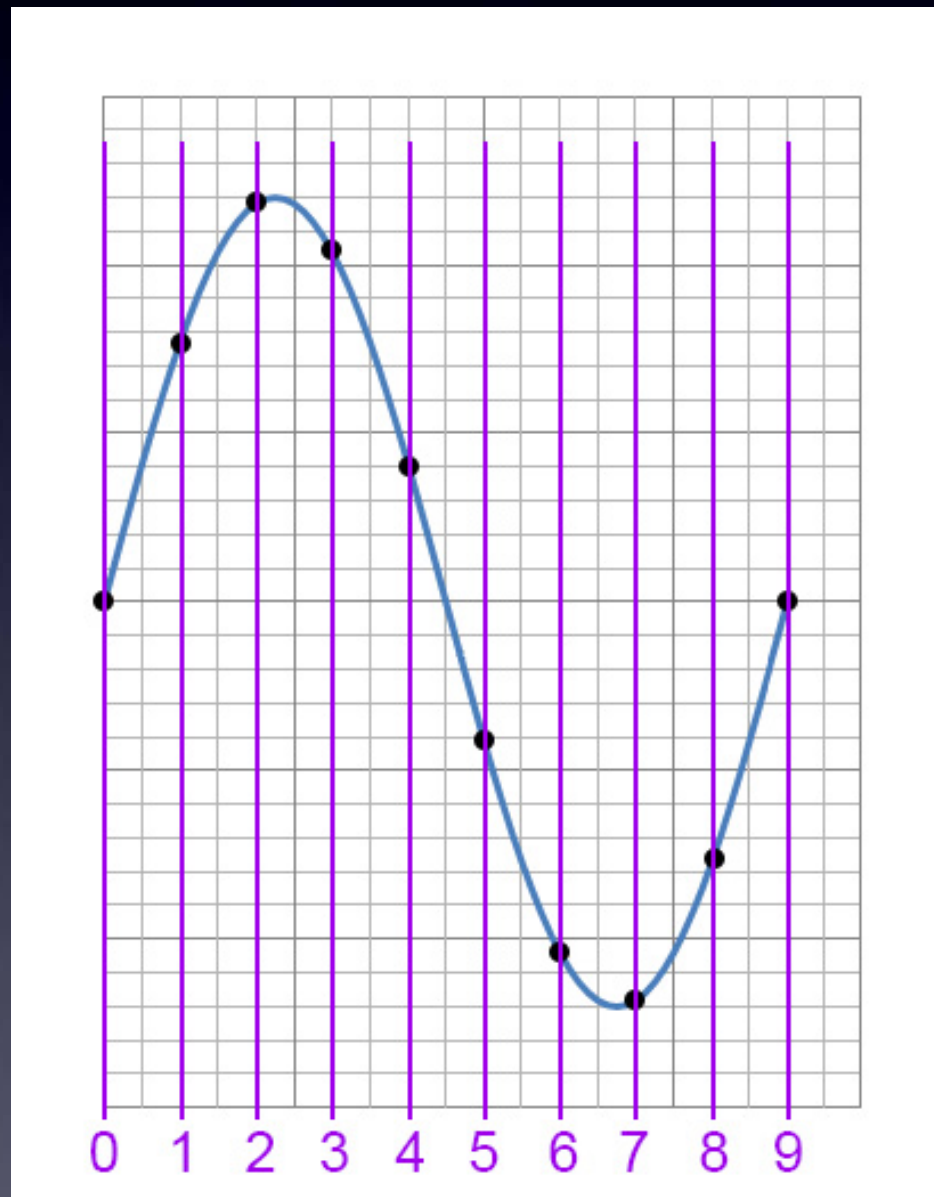


# Digital Signal Processing

To record it digitally

First slice it

Then get the values

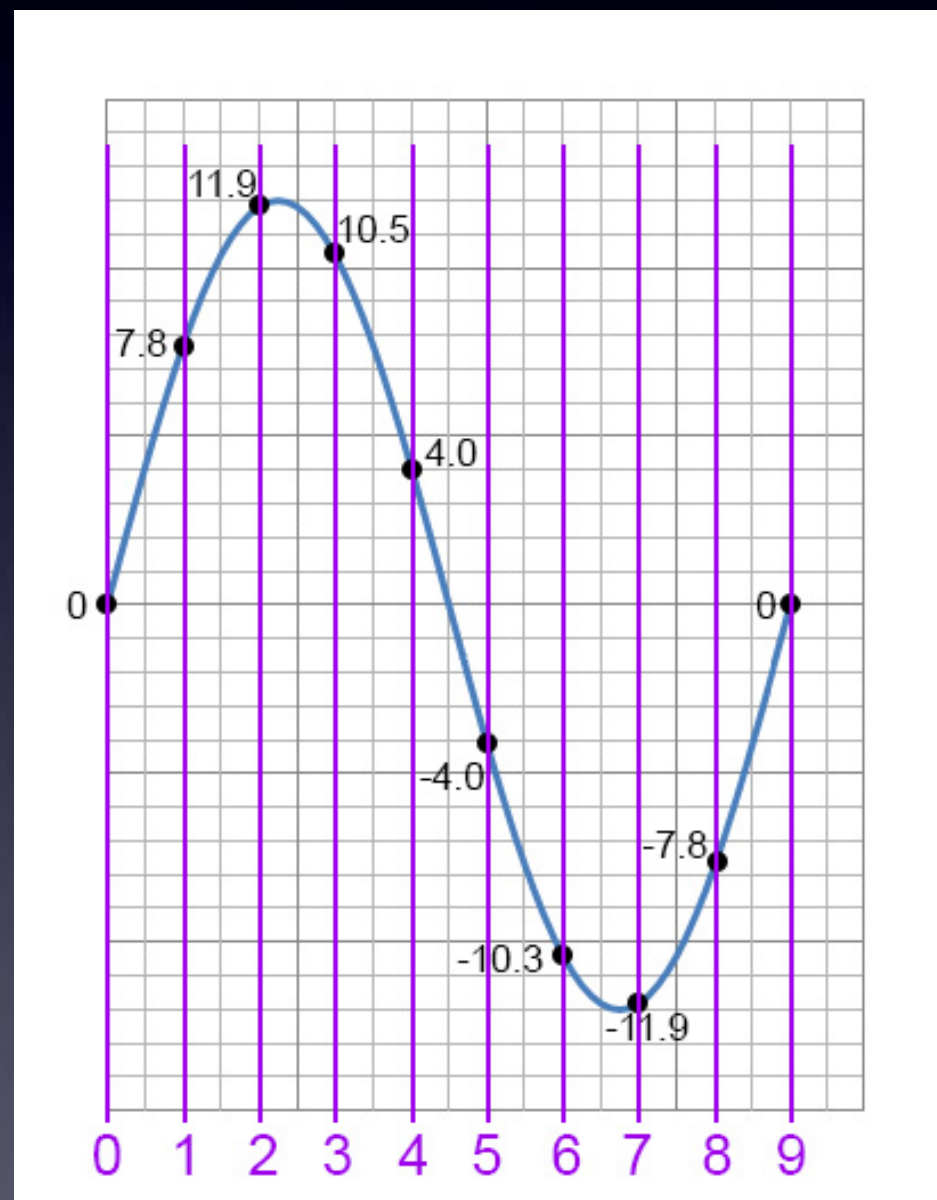


# Digital Signal Processing

To record it digitally

First slice it

Then get the values



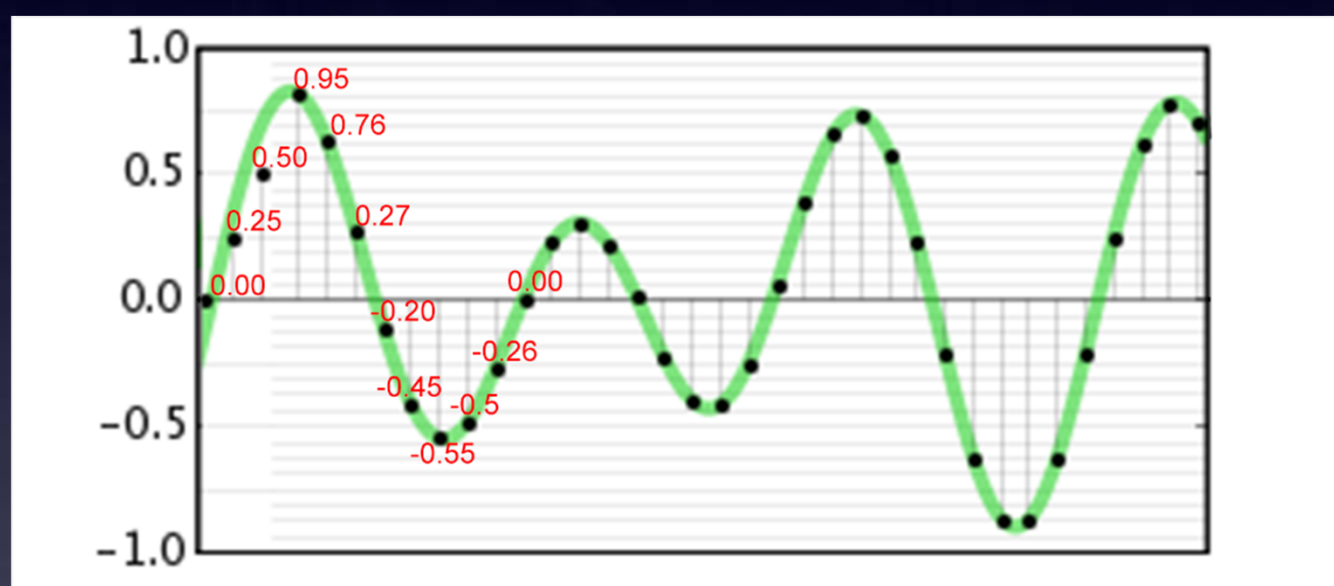








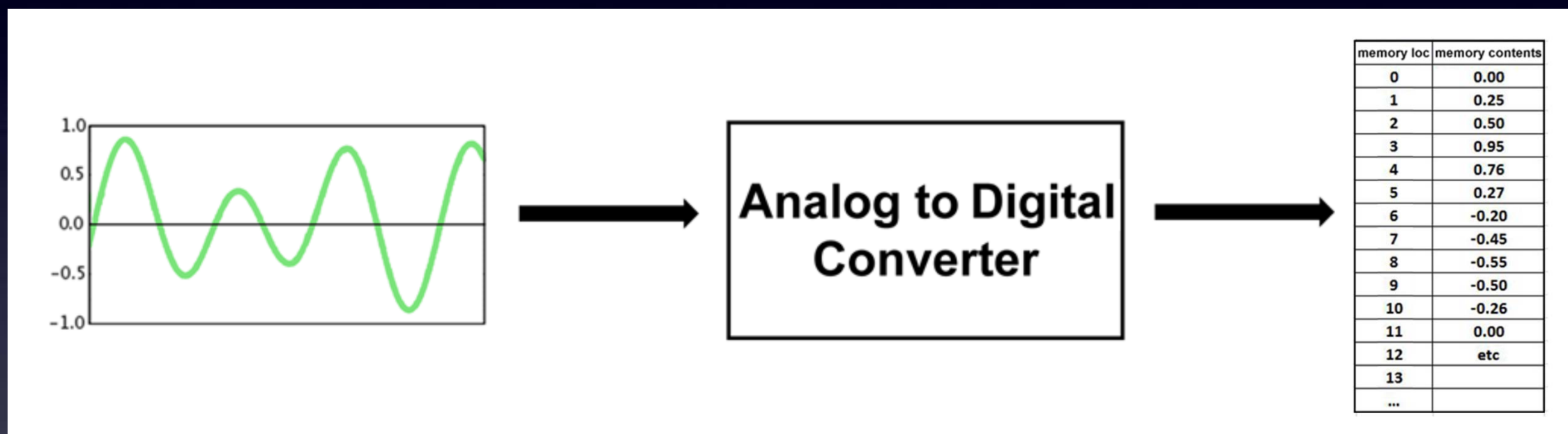
# Digital Signal Processing



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

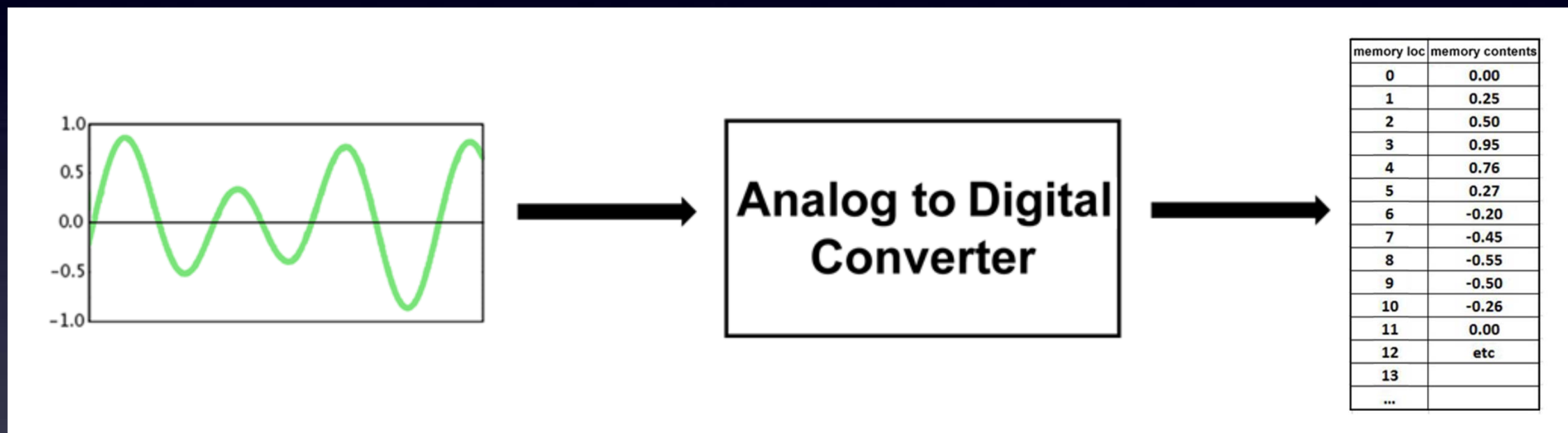
# Digital Signal Processing



## Analog to Digital Conversion:

sampling an analog waveform to store it in digital memory

# Digital Signal Processing



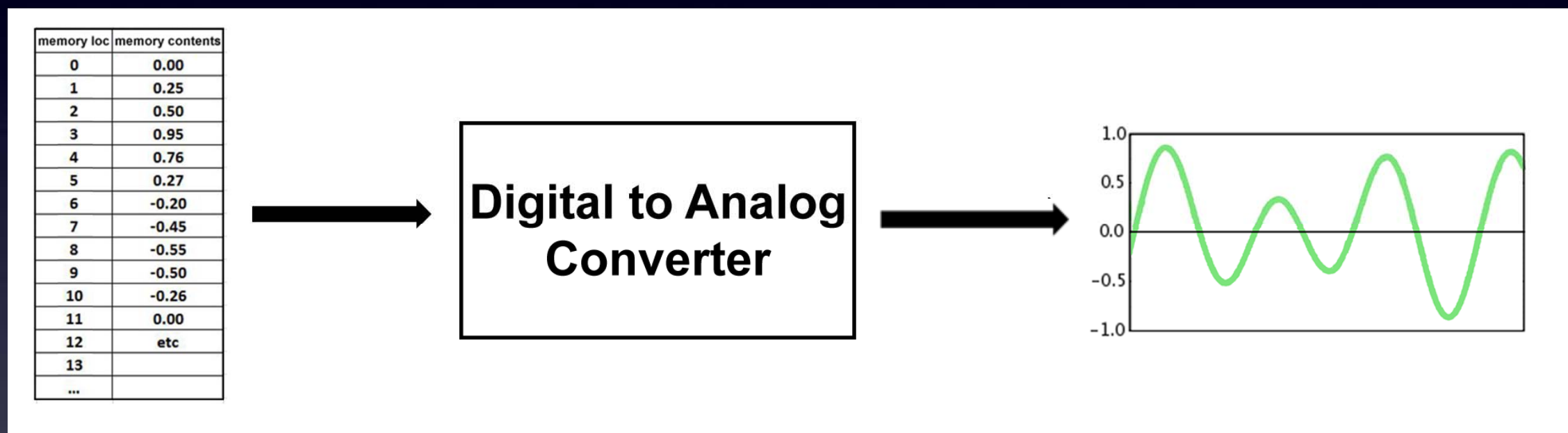
**A/D**

sampling an analog waveform to store it in digital memory

# Digital Signal Processing

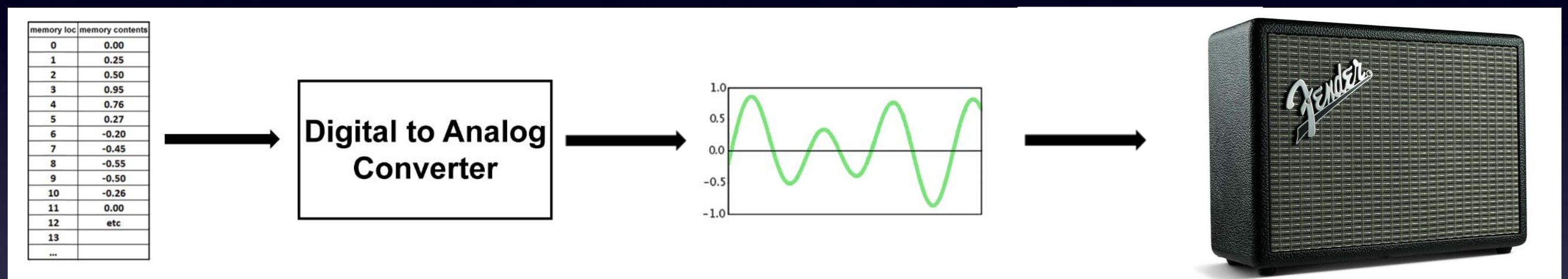
**How do we play back a digitized waveform?**

# Digital Signal Processing



**Digital to Analog Conversion:**  
Playing back the Digitized waveform

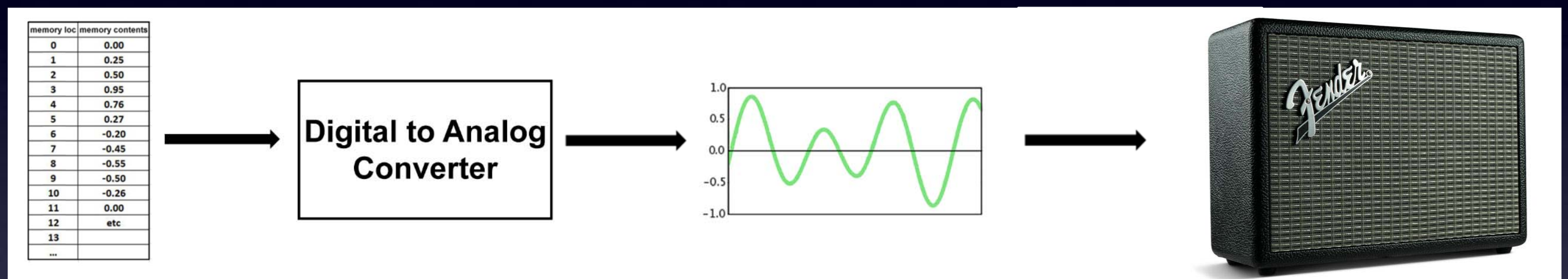
# Digital Signal Processing



**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 ?



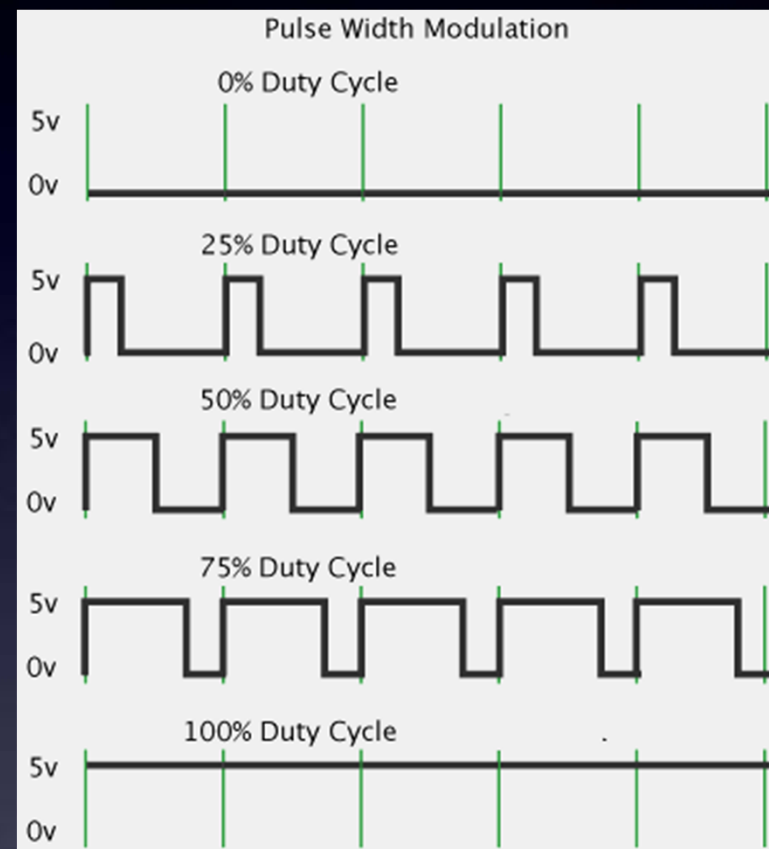
**Pulse Wave:**

ON and OFF at any ratio you like

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

# Digital Signal Processing

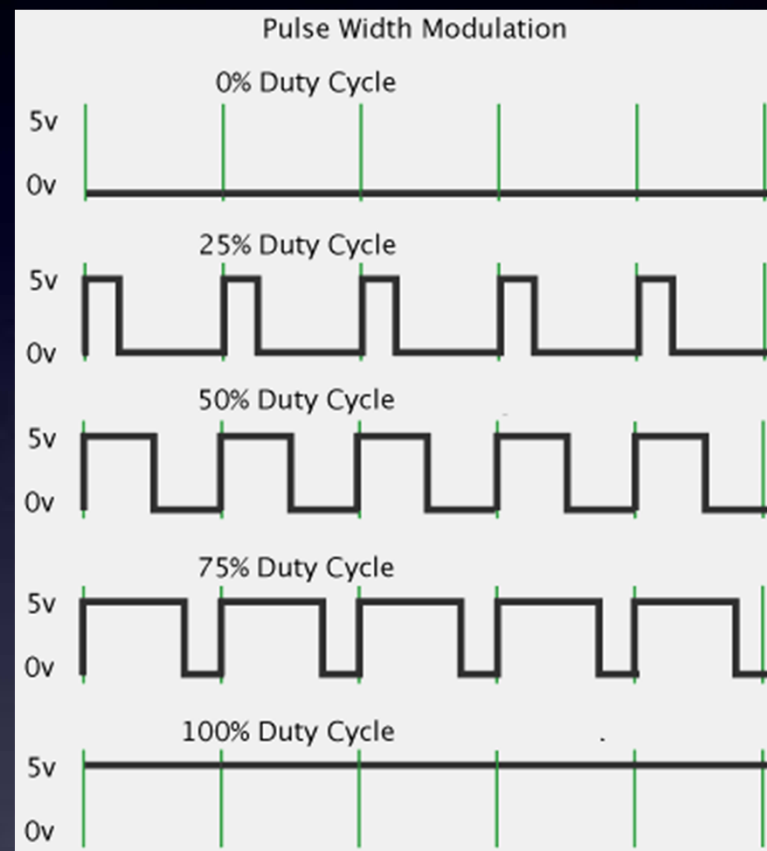
PWM ?



**Pulse Wave:**

ON and OFF at any ratio you like

# Digital Signal Processing

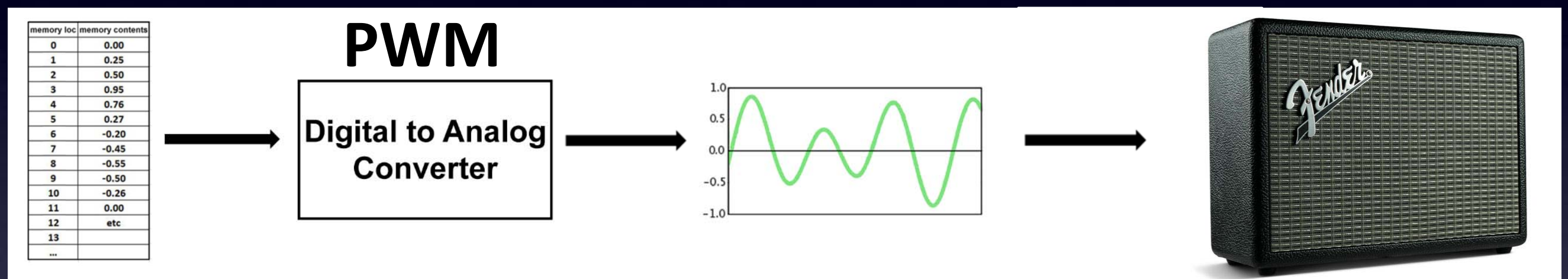


**PWM**

**Pulse Width Modulation**



# Digital Signal Processing



D/A

Using PWM for playing back the Digitized waveform

# Digital Signal Processing

Kind of complicated to code

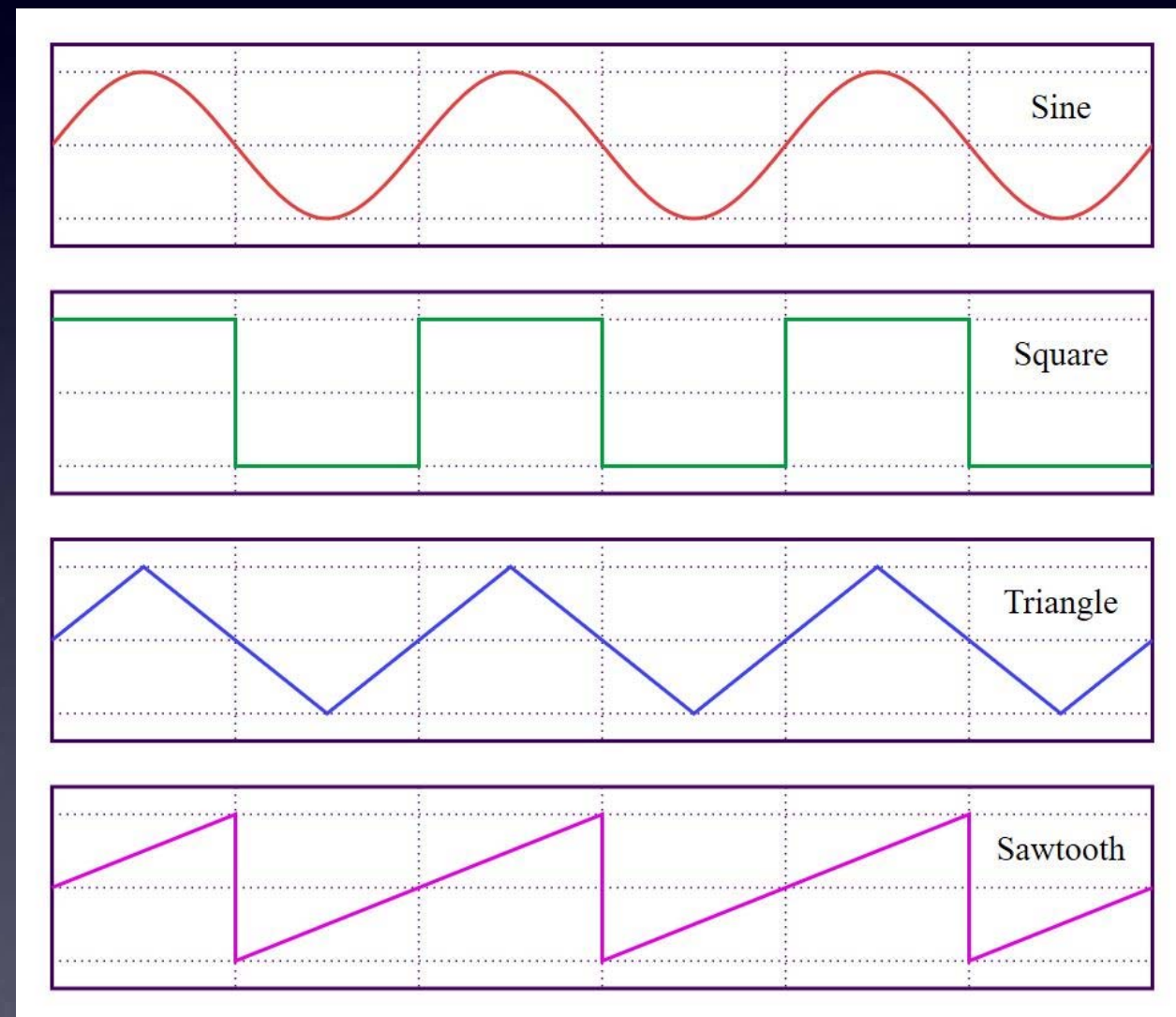
*So, my ArduTouch software makes it easy*

- 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”:



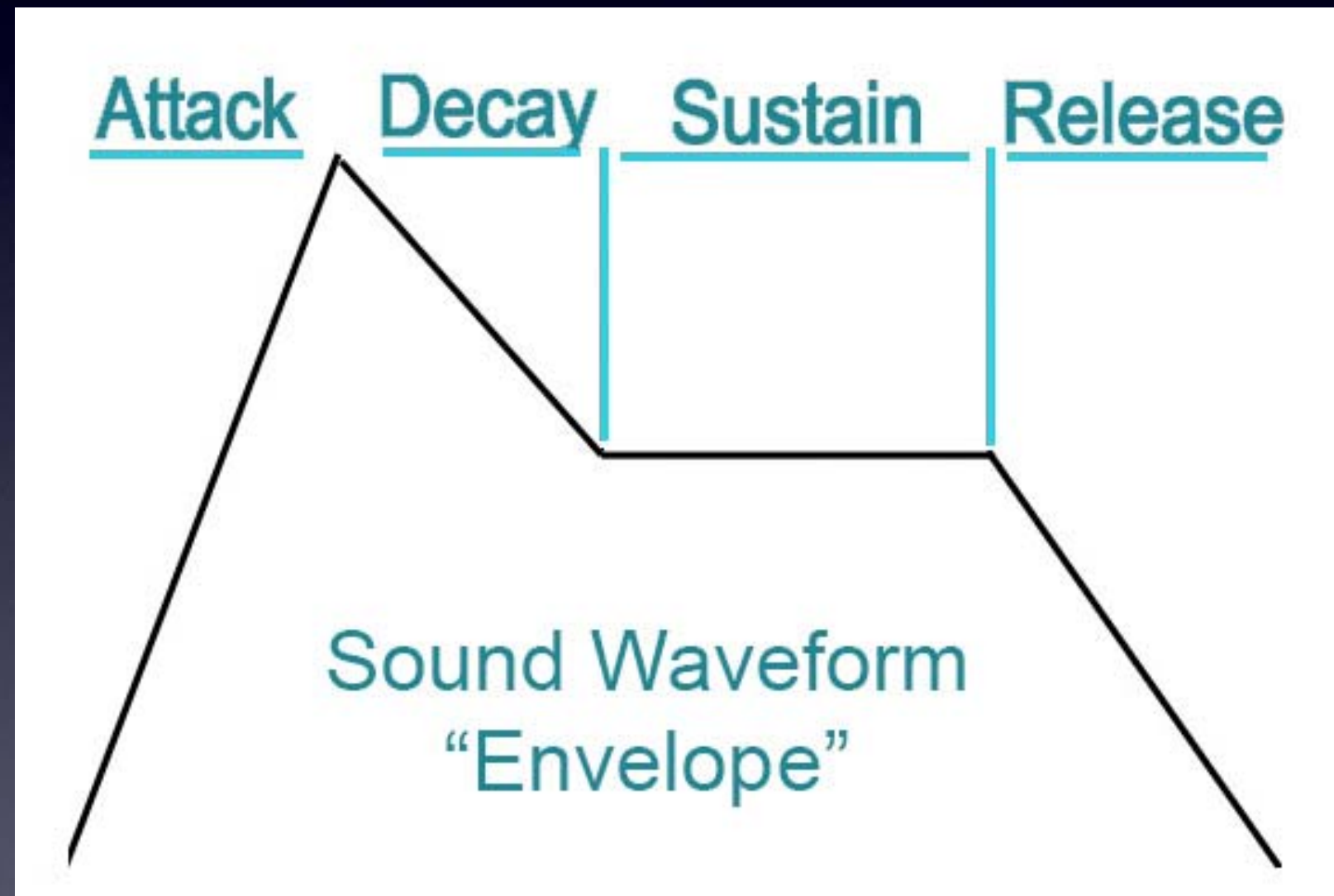
# Digital Signal Processing

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*

```
ArduTouch
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

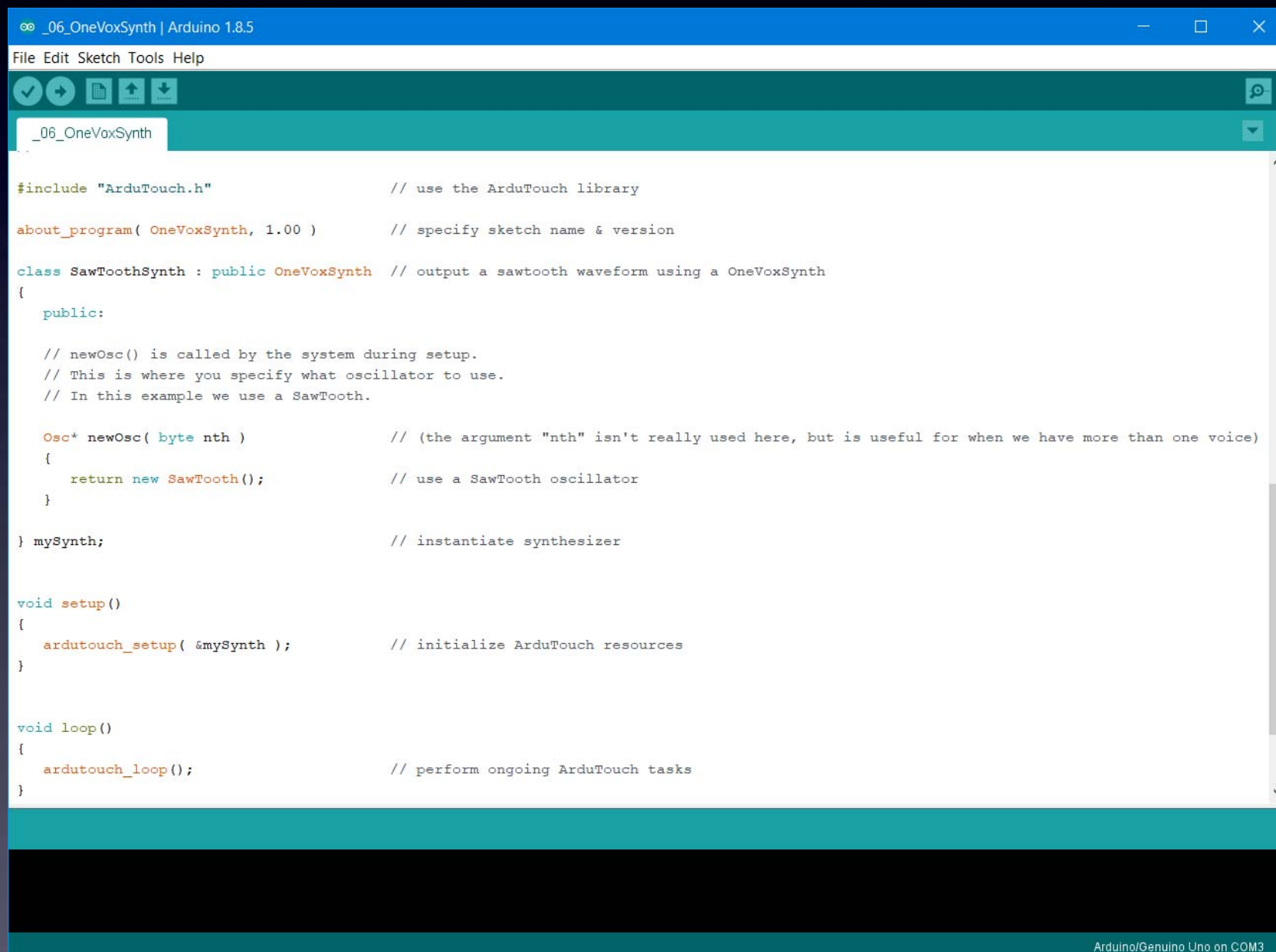
# ArduTouch *Arduino-Compatible*

```
_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*



```
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
```

**With  
extensive  
Arduino library  
for ArduTouch**

*to make it easy  
to create  
your own synths*

# ArduTouch

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

#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
```

## Complete code for:

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

# ArduTouch

```
Arduino IDE window: _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



The screenshot shows the GitHub repository page for 'maltman23/ArduTouch'. The browser address bar shows the URL 'https://github.com/maltman23/ArduTouch'. The repository name is 'maltman23 / ArduTouch' with 4 Unwatch, 9 Star, and 0 Fork buttons. The repository description states: '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'. The repository statistics show 7 commits, 1 branch, 0 releases, and 1 contributor. The file list includes: 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), and README.md (Create README.md, 7 months ago). The latest commit is b5109e5 on Jul 10.

Open Hardware – everything is on Github  
maltman23

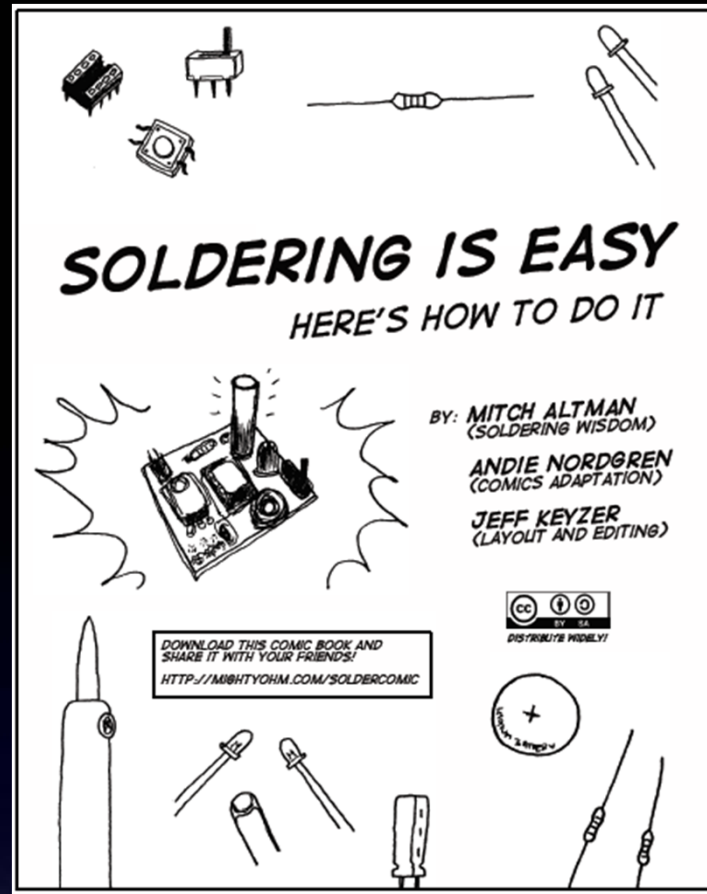


(Don't bring these home)

# 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.)

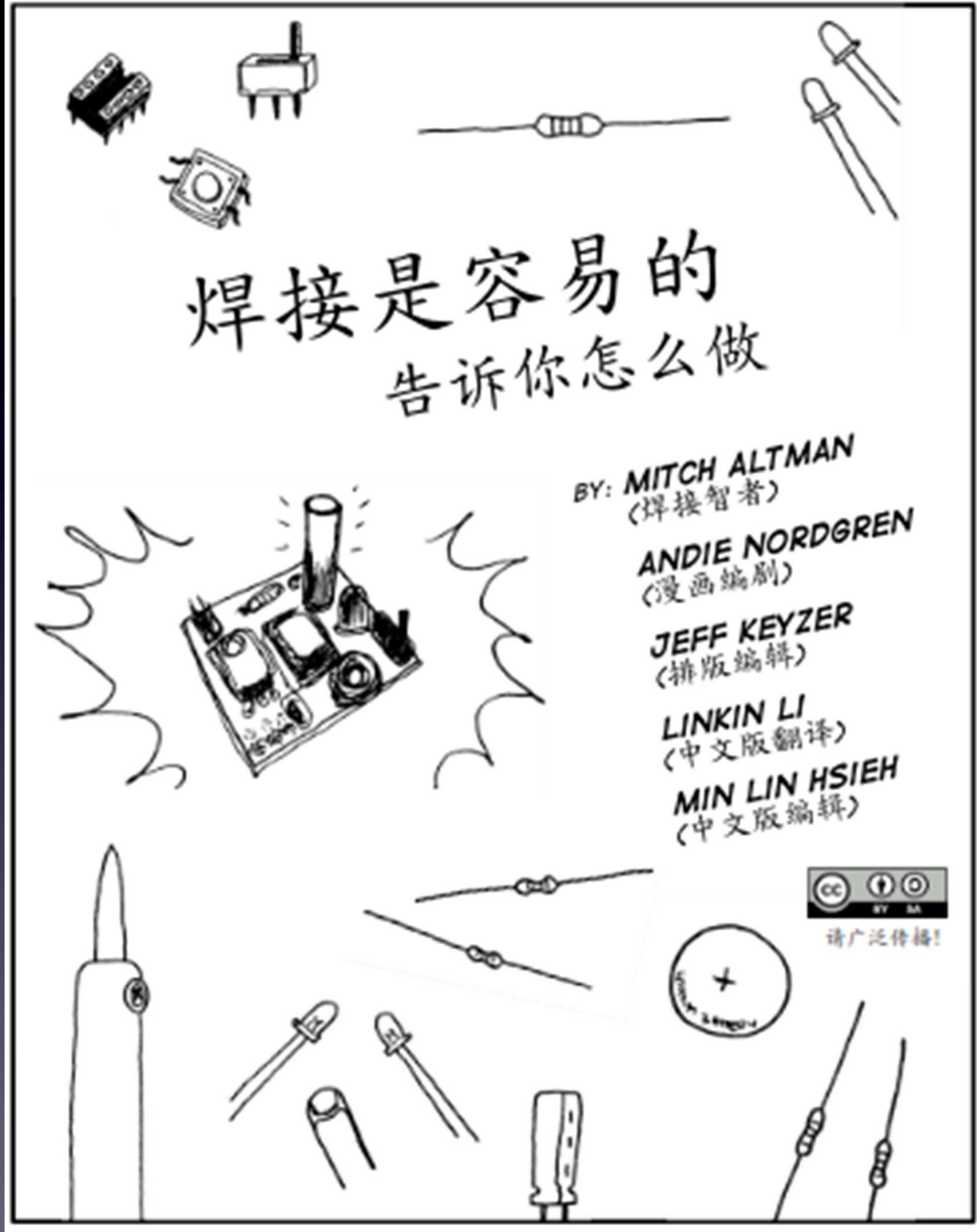
download for free at:  
<http://mightyohm.com/soldercomic>

# Learn To Solder



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

# Learn To Solder



Download in the language of your choice for free at:  
<http://mightyohm.com/soldercomic>

# Learn To Solder

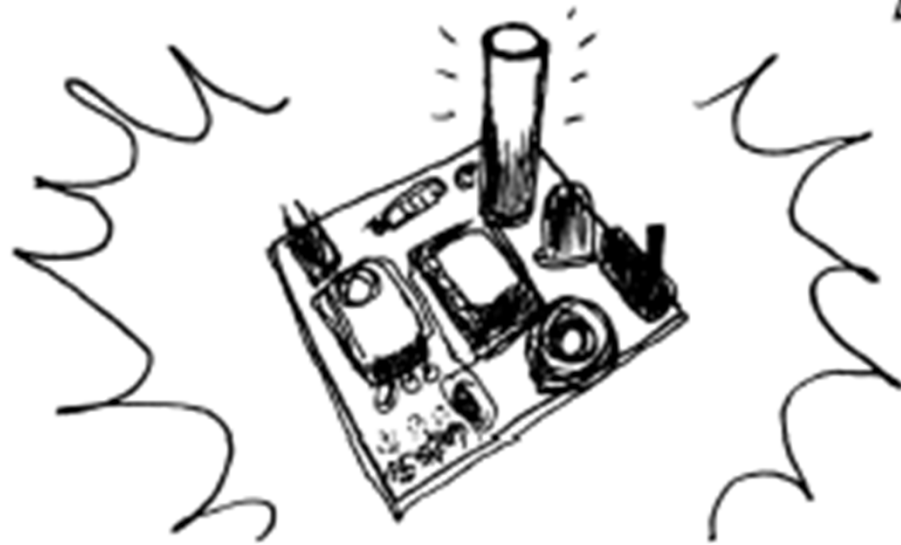
## ***SOLDER C'EST FACILE*** ***VOICI COMMENT FAIRE***

DE: ***MITCH ALTMAN***  
(MAITRE SOUDEUR)

***ANDIE NORDGREN***  
(ADAPTATION BD)

***JEFF KEYZER***  
(EDITION, MISE EN PAGE)

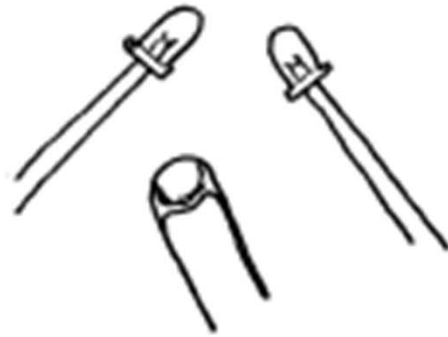
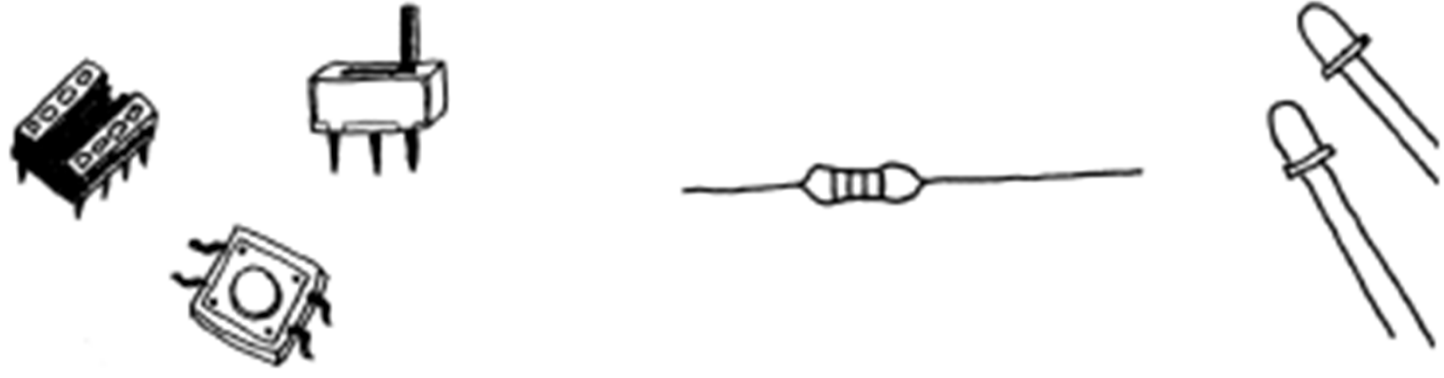
***SNOOTLAB***  
(TRADUCTION FR.)



TELECHARGEZ CETTE BD  
ET PARTAGEZ LA AVEC VOS AMIS !  
[HTTP://MIGHTYOHM.COM/SOLDERCOMIC](http://mightyohm.com/soldercomic)



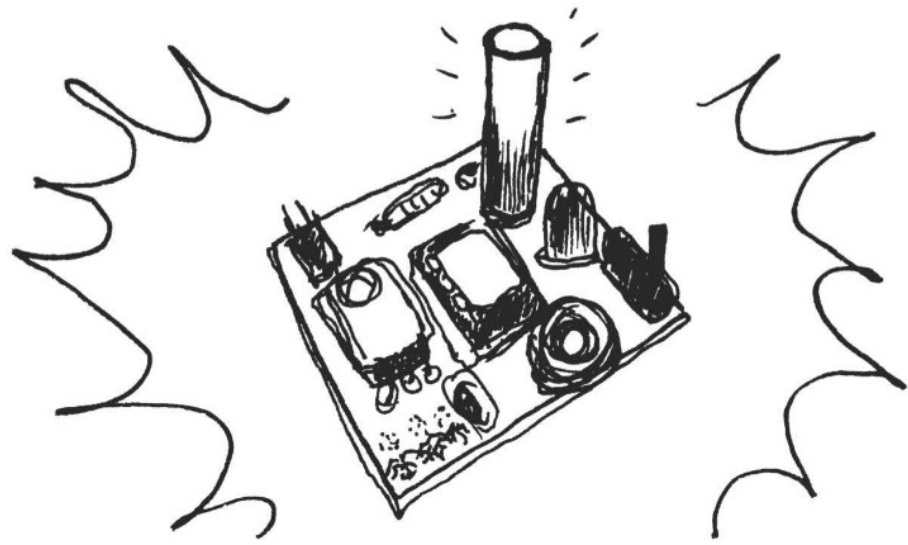
A DIFFUSER LARGEMENT !



Download in the language of your choice for free at:  
<http://mightyohm.com/soldercomic>

# Learn To Solder

## **SOLDAR ES FÁCIL!** *APRENDE CÓMO HACERLO*



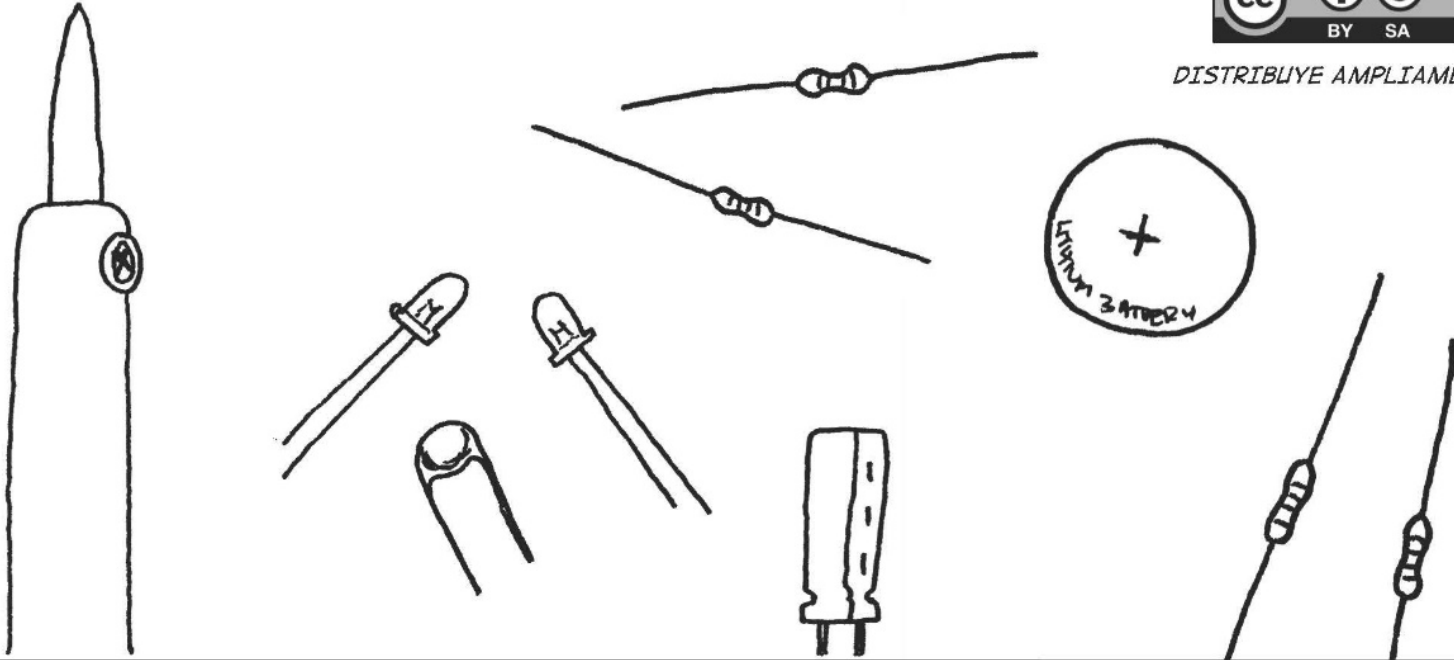
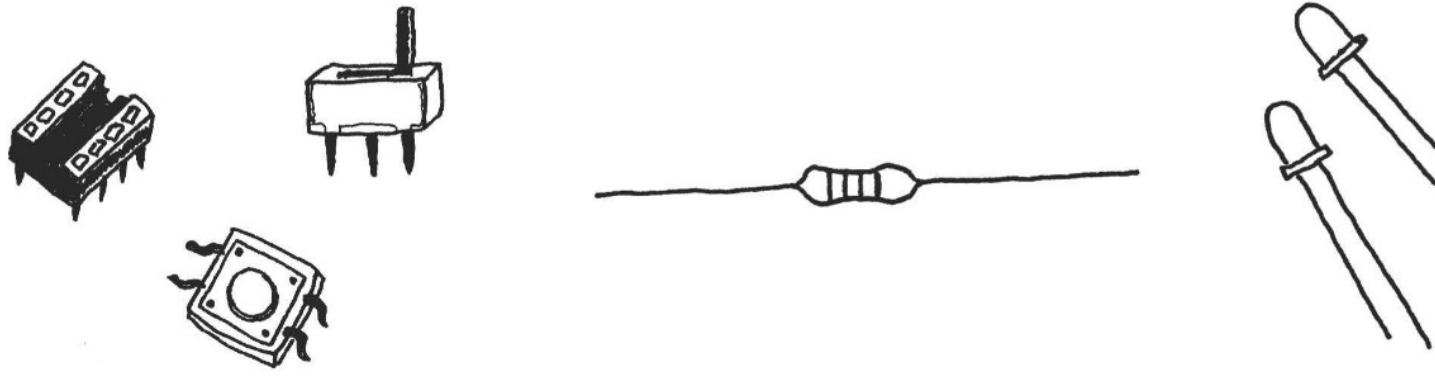
**POR: MITCH ALTMAN**  
(SABIDURÍA EN SOLDADO)

**ANDIE NORDGREN**  
(ADAPTACIÓN A COMIC)

**JEFF KEYZER**  
(DISEÑO Y EDICIÓN)



*DISTRIBUYE AMPLIAMENTE!*

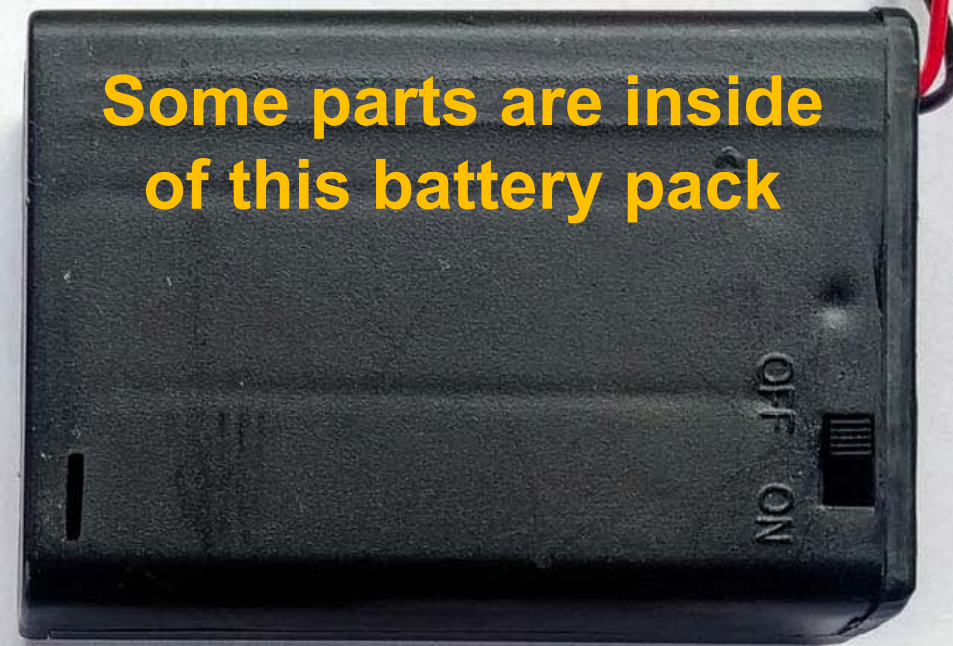
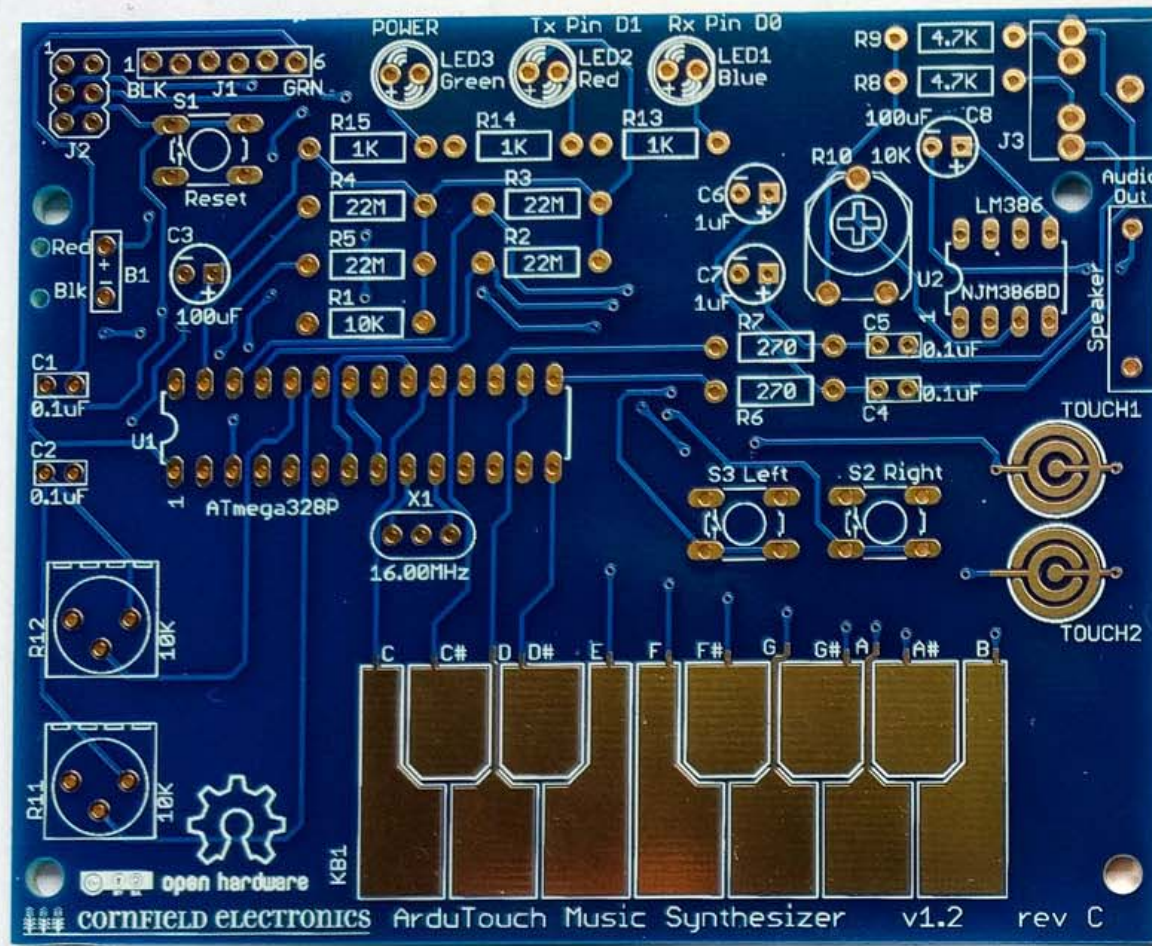


Download in the language of your choice for free at:  
<http://mightyohm.com/soldercomic>

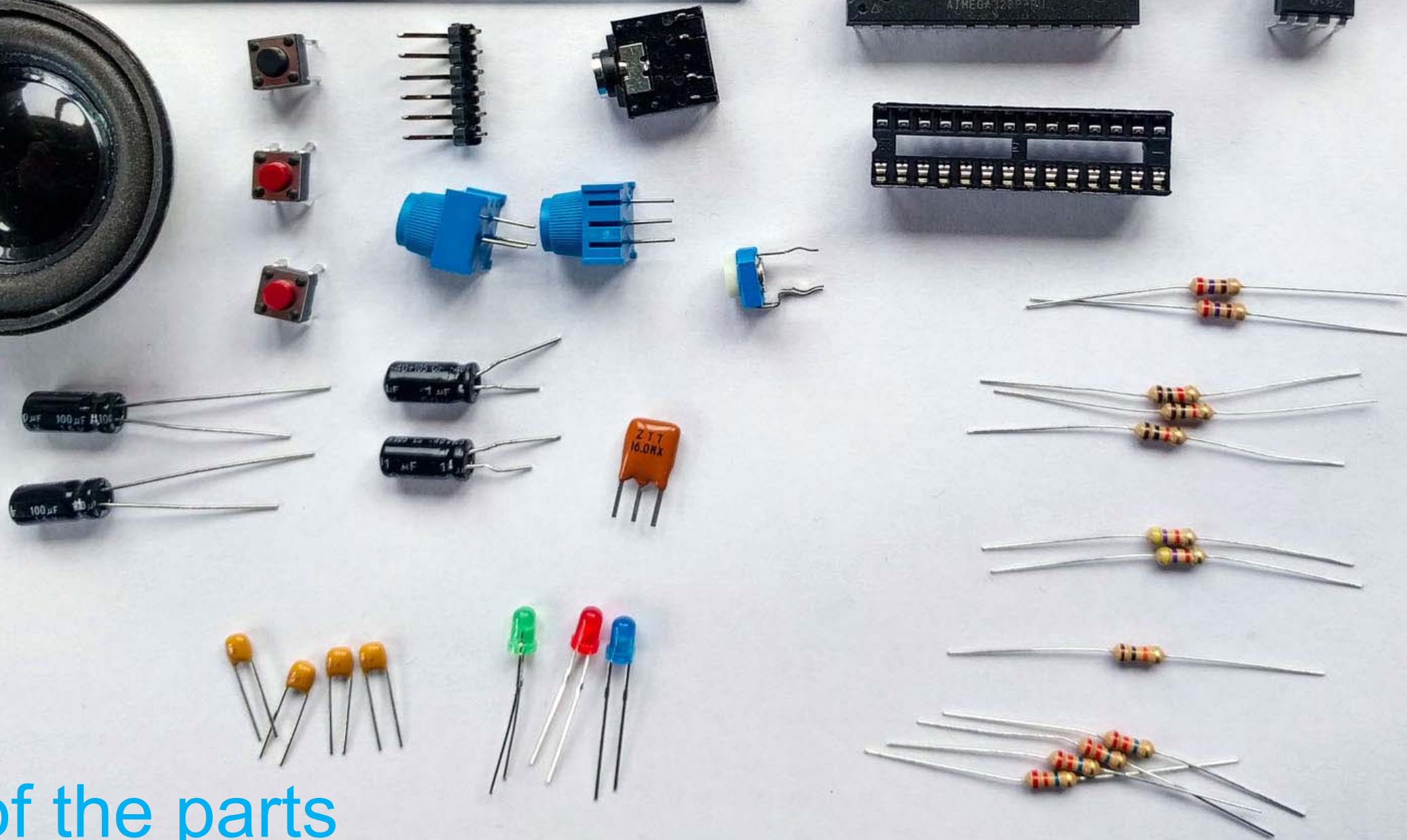
# Learn To Solder



Download in the language of your choice for free at:  
<http://mightyohm.com/soldercomic>

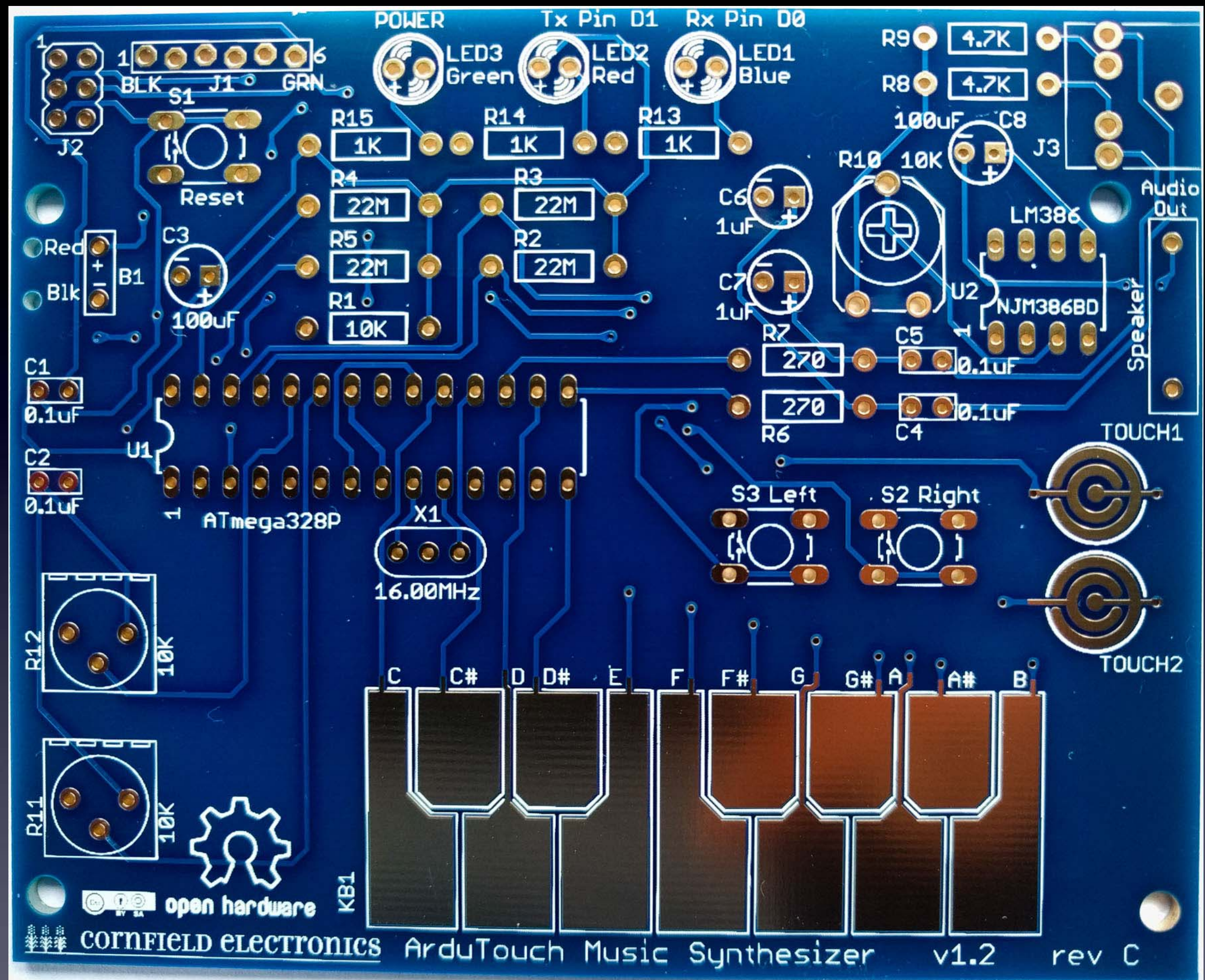


Some parts are inside of this battery pack



All of the parts





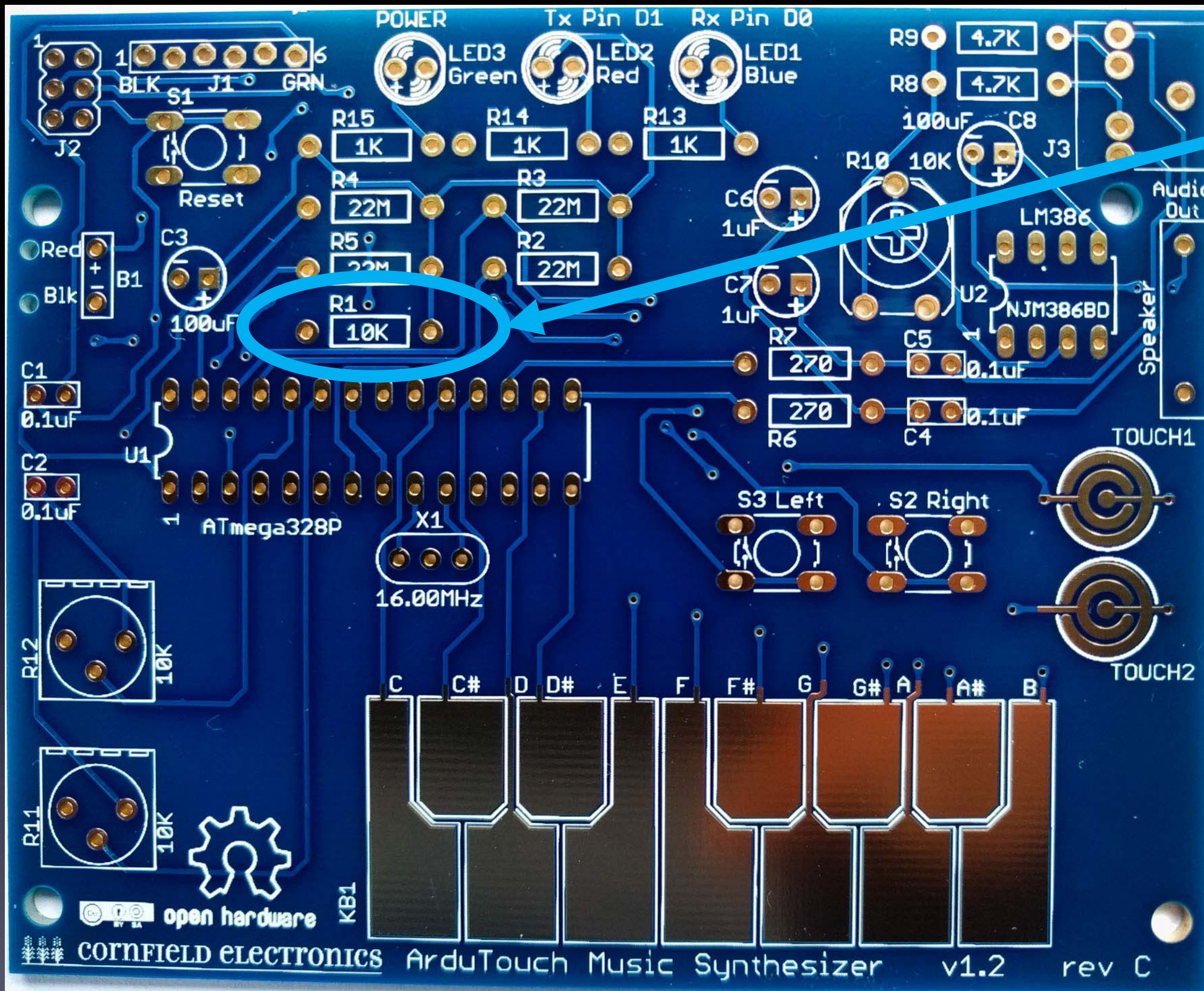
The board we'll solder the parts to



**Important:**  
**Use solder WITH lead (Pb) !!**  
**Unleaded solder**  
**has very poisonous fumes!**

### The tools you'll need:

- soldering Iron (35W or less)
- solder (60/40 Sn/Pb, rosin core, 0.031" diameter or less)
- soldering iron stand
- cellulose kitchen sponge (*not plastic!*)
- *small* wire cutter

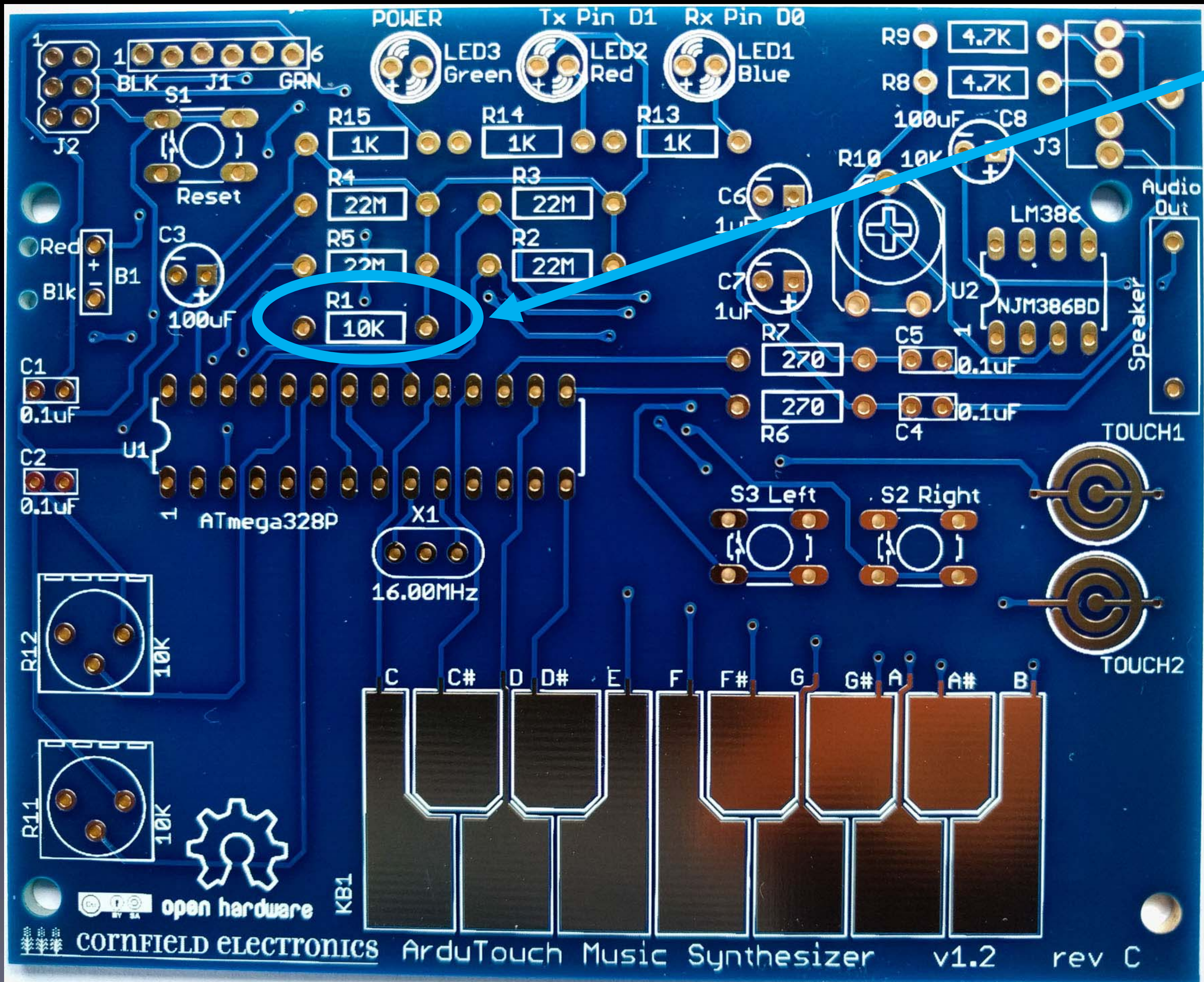


**R1 – this is where it goes**



**R1: Brown, Black, Orange**

**(not Brown, Black, Red)**



R1

POWER Tx Pin D1 Rx Pin D0

LED3 Green LED2 Red LED1 Blue

R1 10K

Reset

ATmega328P

X1 16.00MHz

S3 Left

S2 Right

TOUCH1

TOUCH2

Audio Out

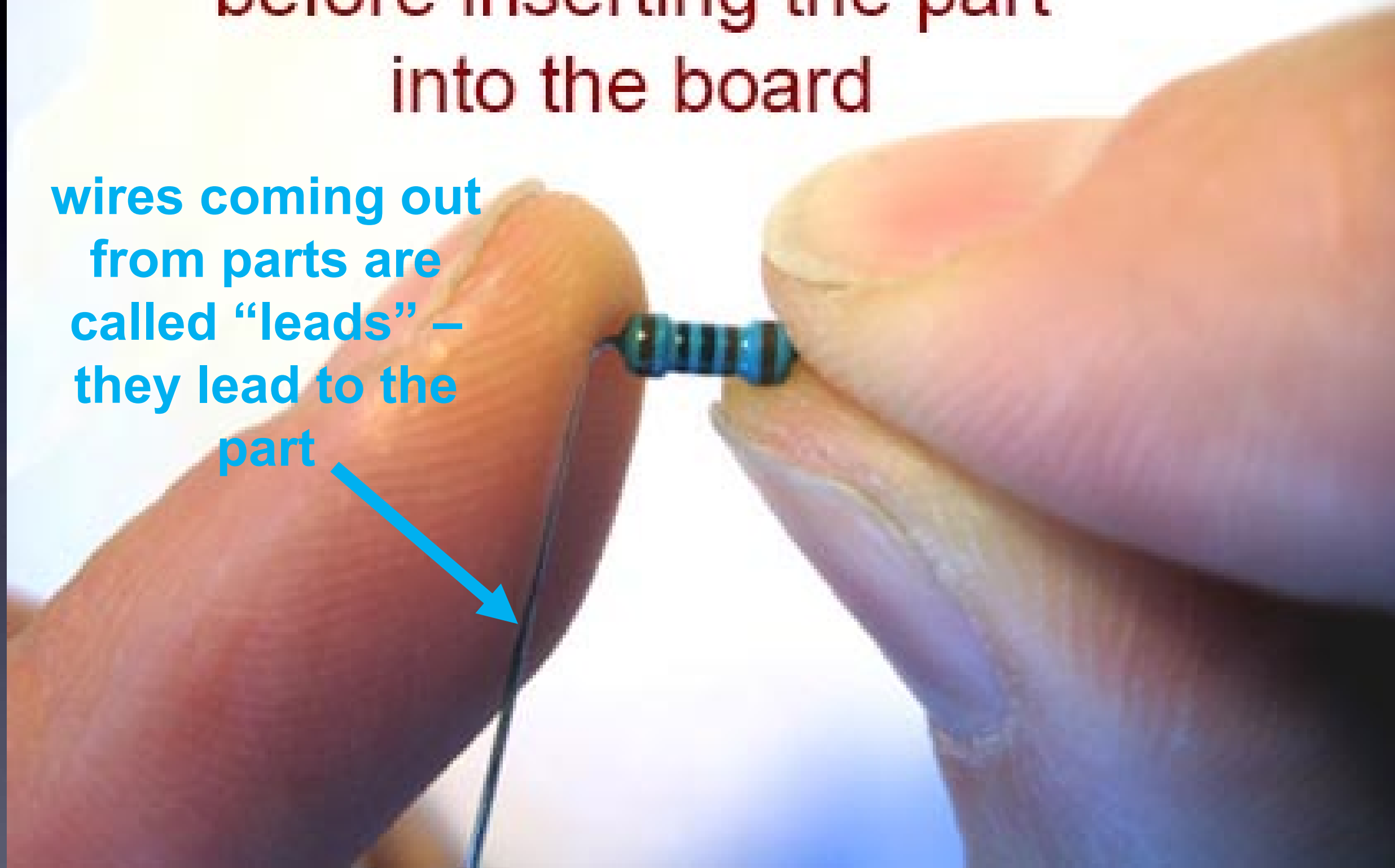
Speaker

open hardware

CORNFIELD ELECTRONICS ArduTouch Music Synthesizer v1.2 rev C

**Bend leads  
before inserting the part  
into the board**

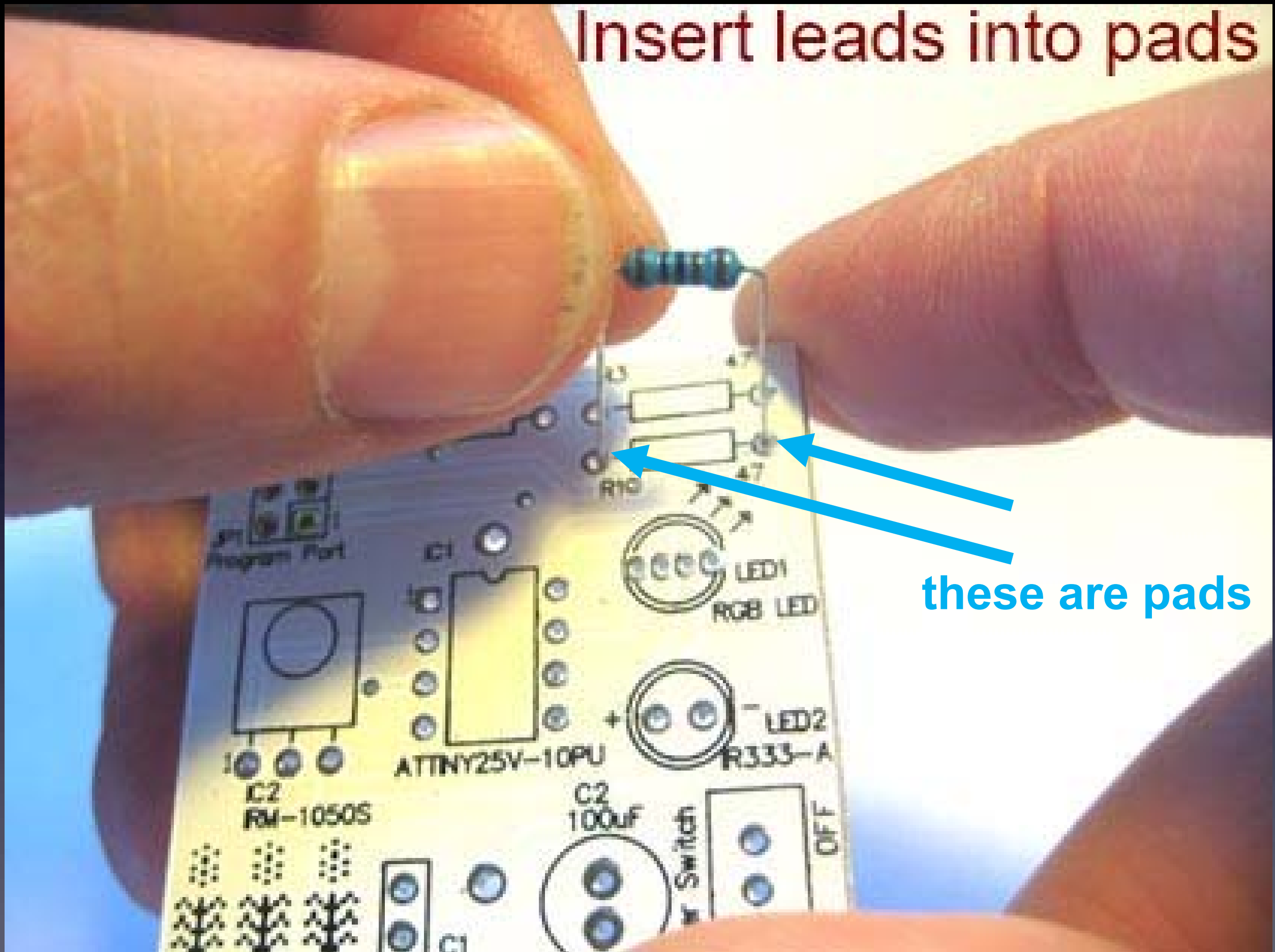
**wires coming out  
from parts are  
called "leads" –  
they lead to the  
part**





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

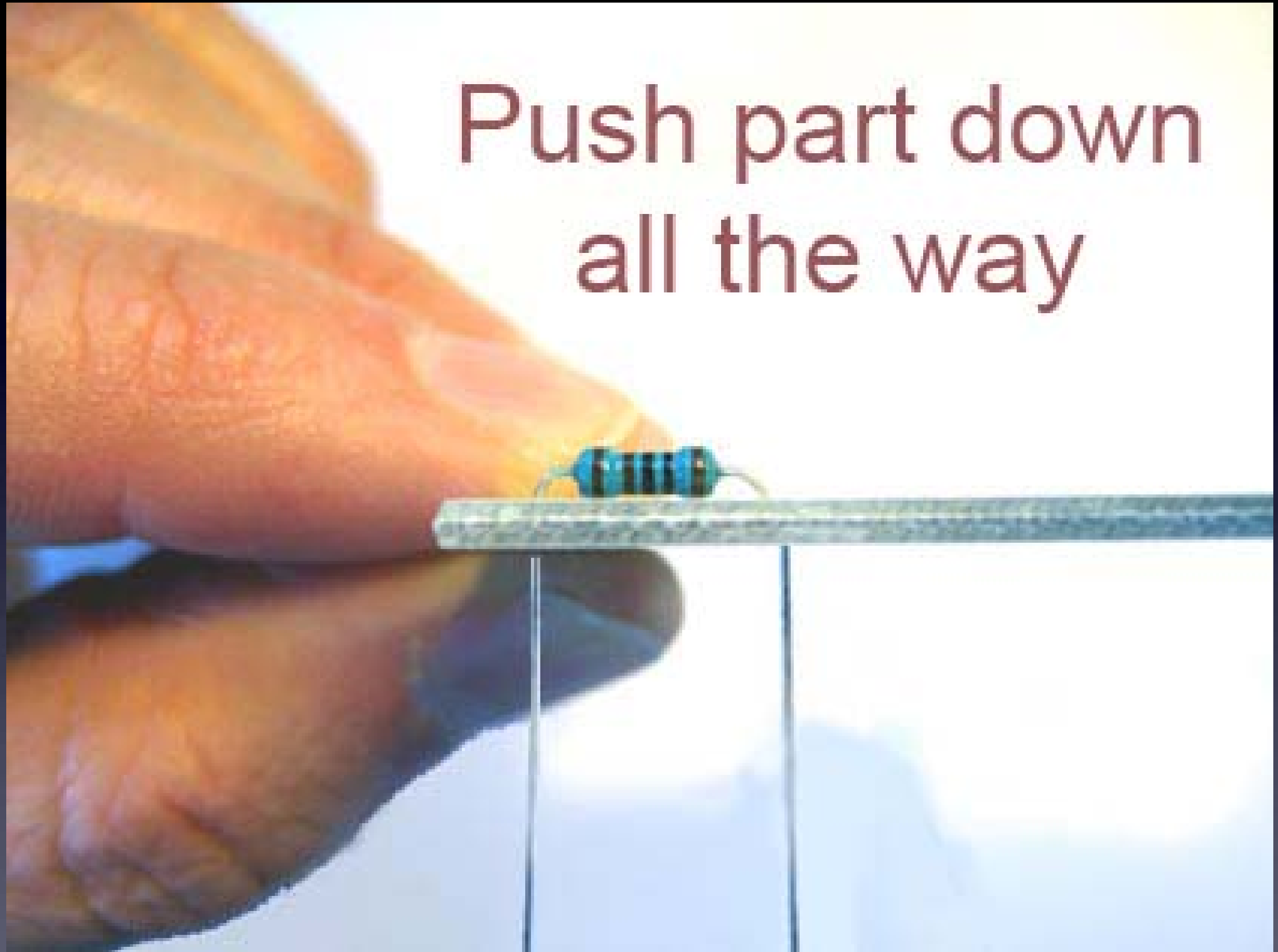
Insert leads into pads

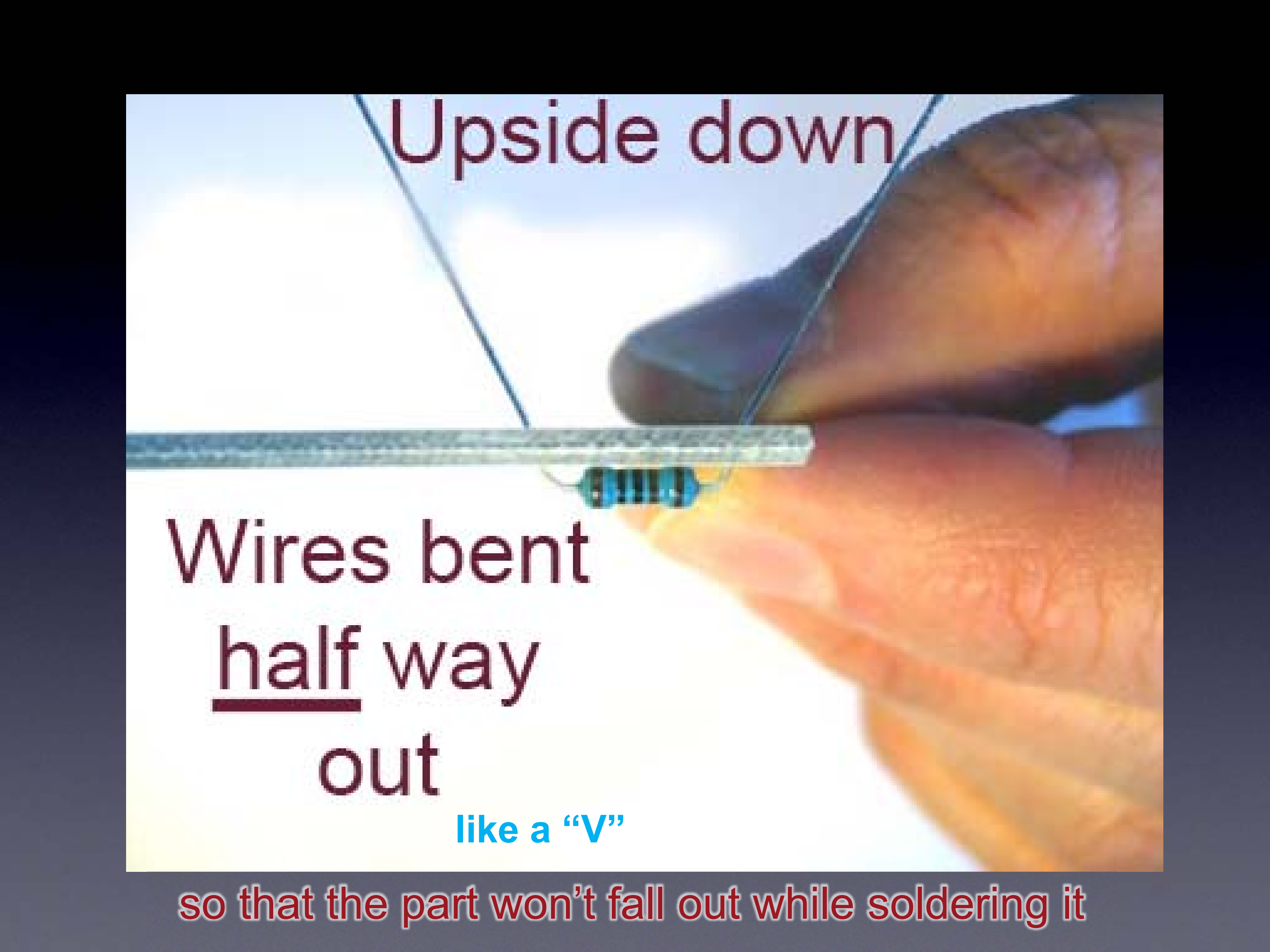


these are pads



Push part down  
all the way



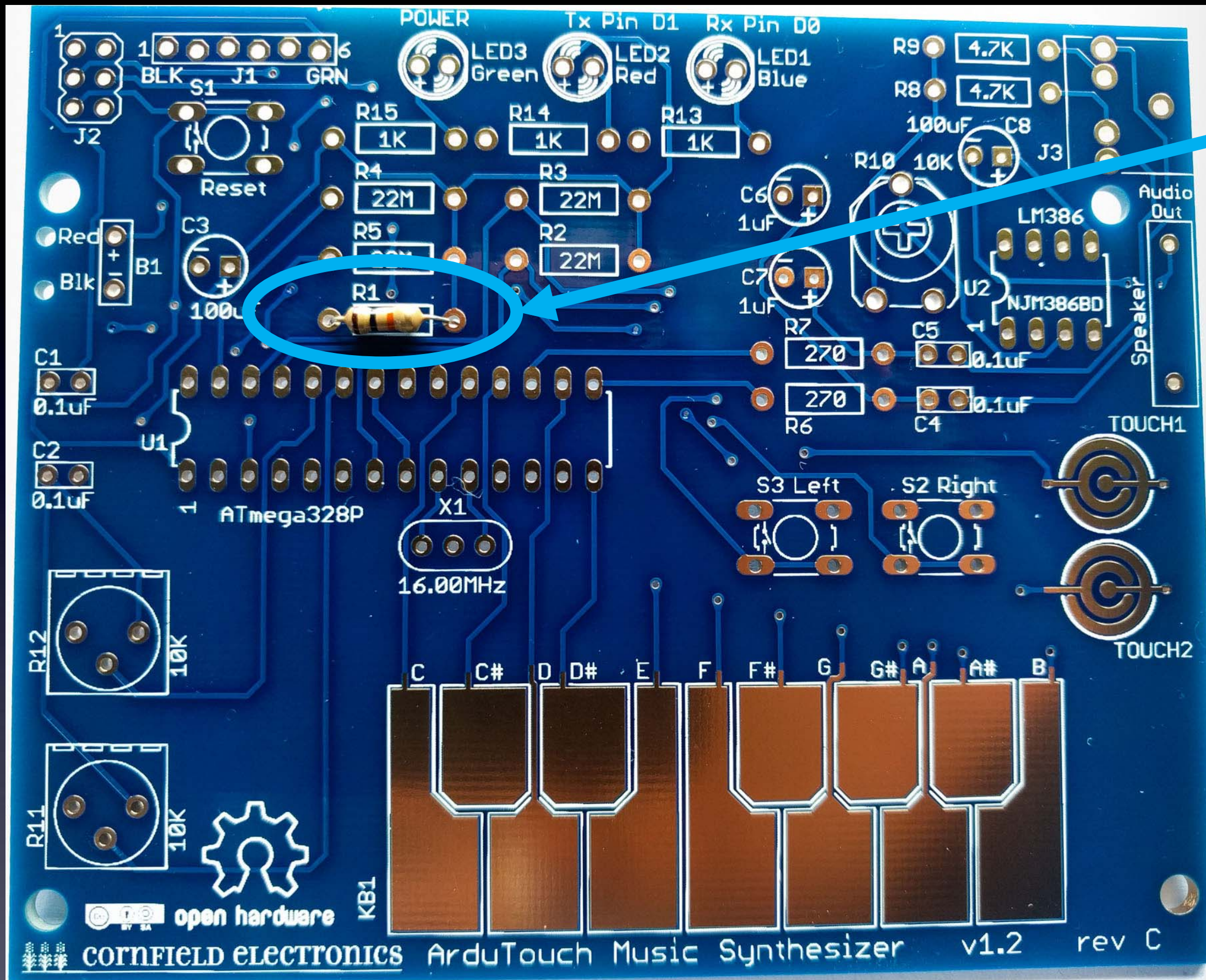


Upside down

Wires bent  
half way  
out

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 iron

(Like a pencil – held from underneath)

**Important**

The perfect kind of  
solder for electronics:

60/40 rosin core,  
0.031" diameter (or smaller)

*(63/37 is also good)*

Important:

Use solder WITH lead (Pb) !!  
lead-free solder  
has very poisonous fumes!

# 3 Safety Tips...

Safety Tip #1:

Hot !!

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

## Safety Tip #2:

Lead (Pb) is toxic

But it easily washes 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)

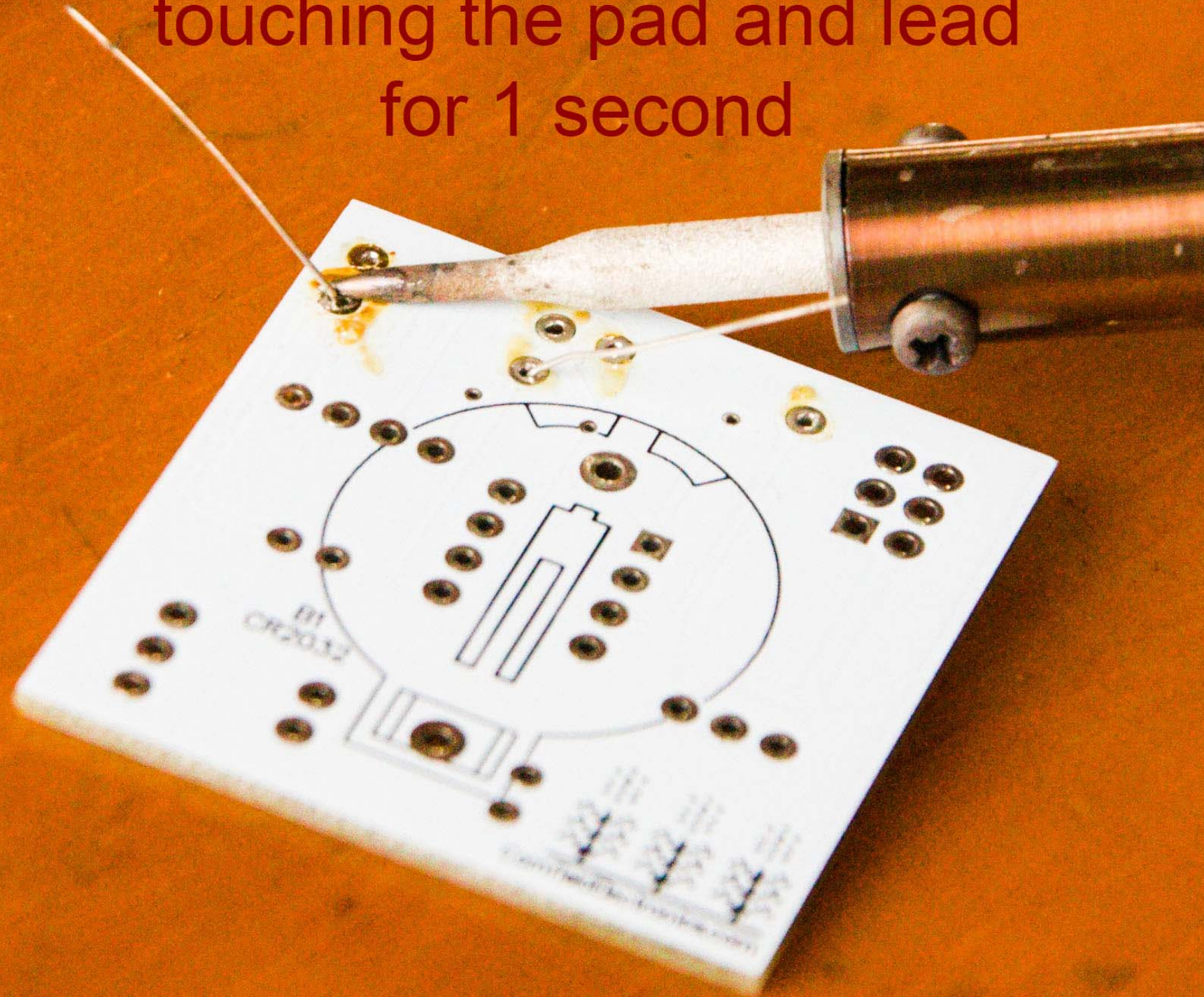
Bang (lightly) 3 times,

Swipe, Rotate, Swipe (on the sponge):

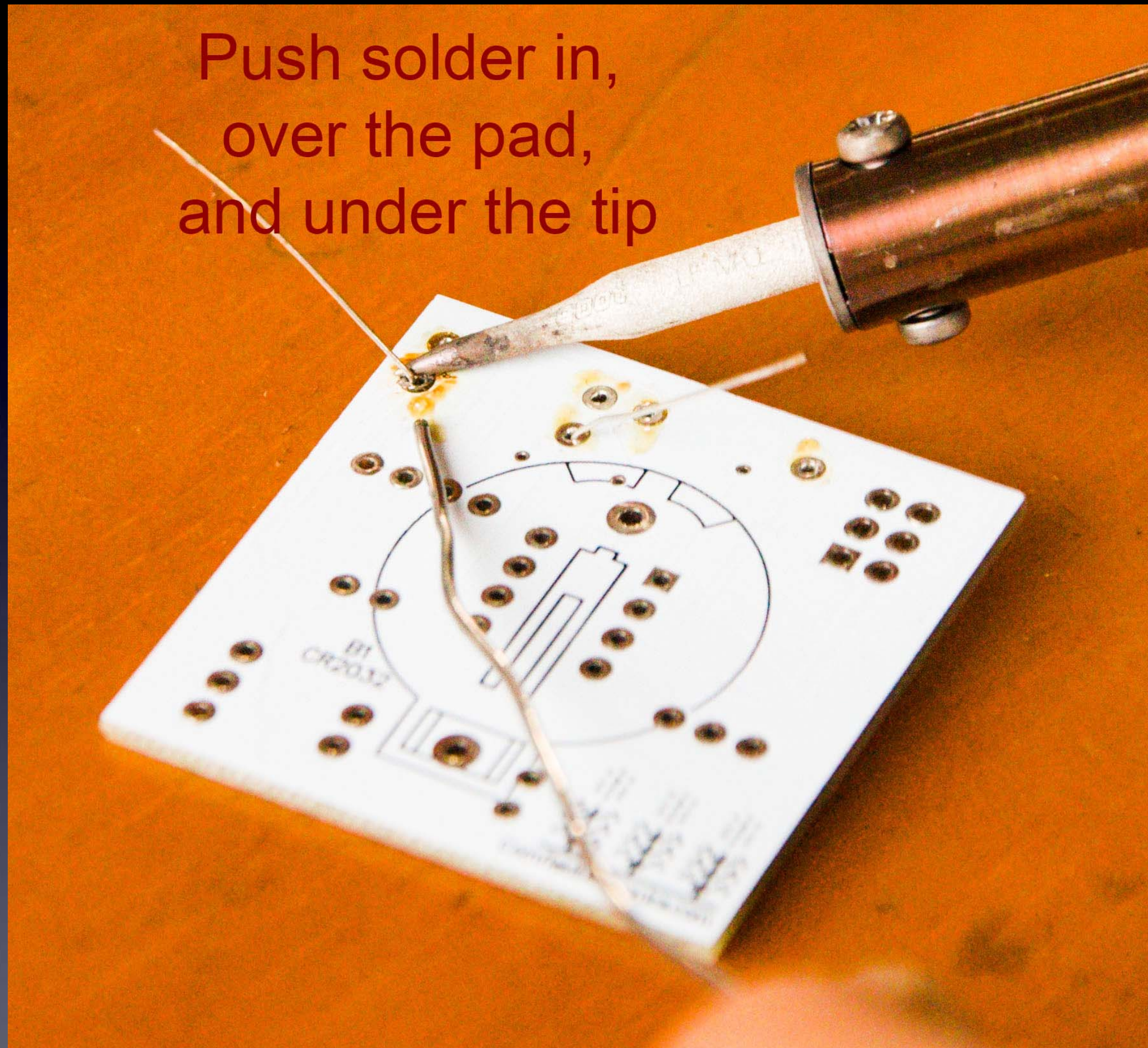
*Keep the tip shiny silver!*

knock solder off the tip

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



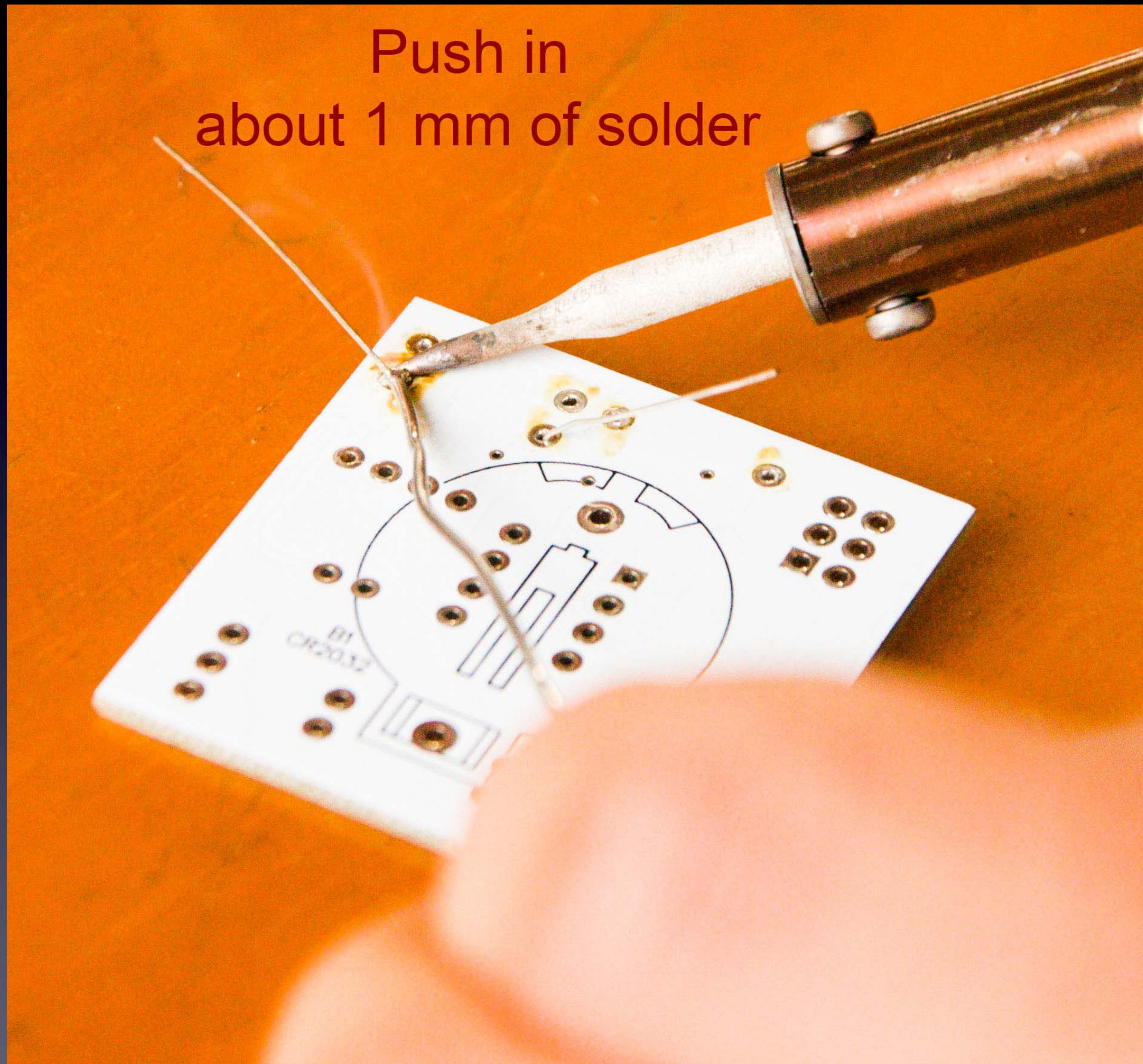
Do this quickly (slowly doesn't work well) – solder in & out in about 1 second



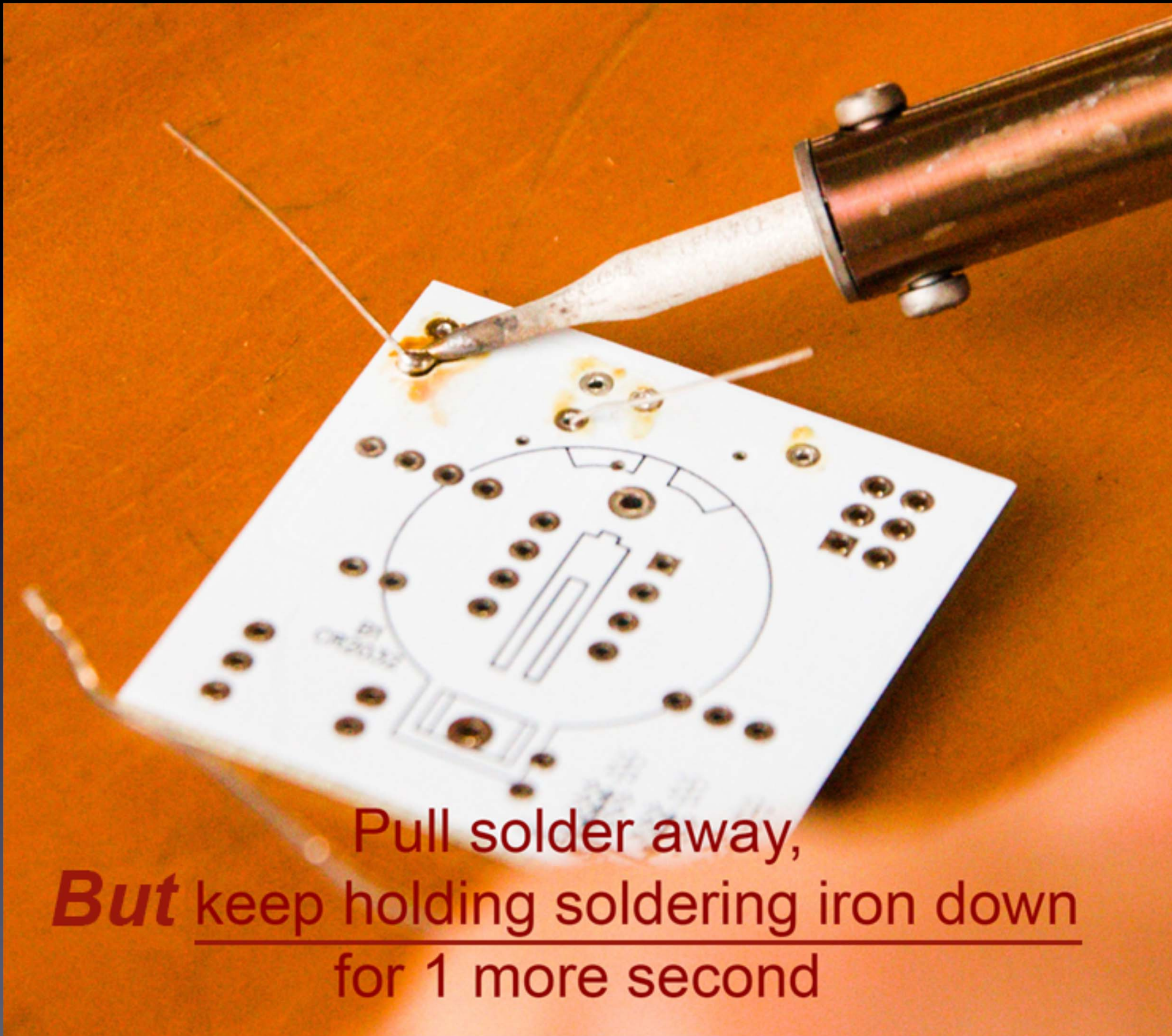
Push solder in,  
over the pad,  
and under the tip

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

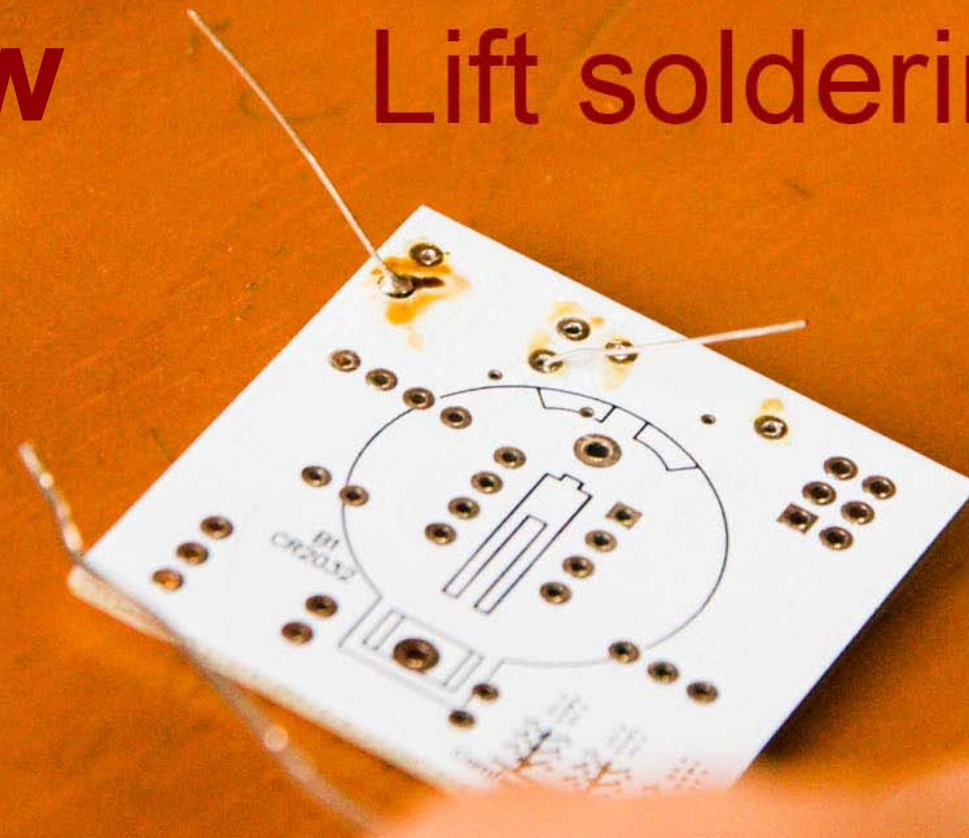
Secret #2:

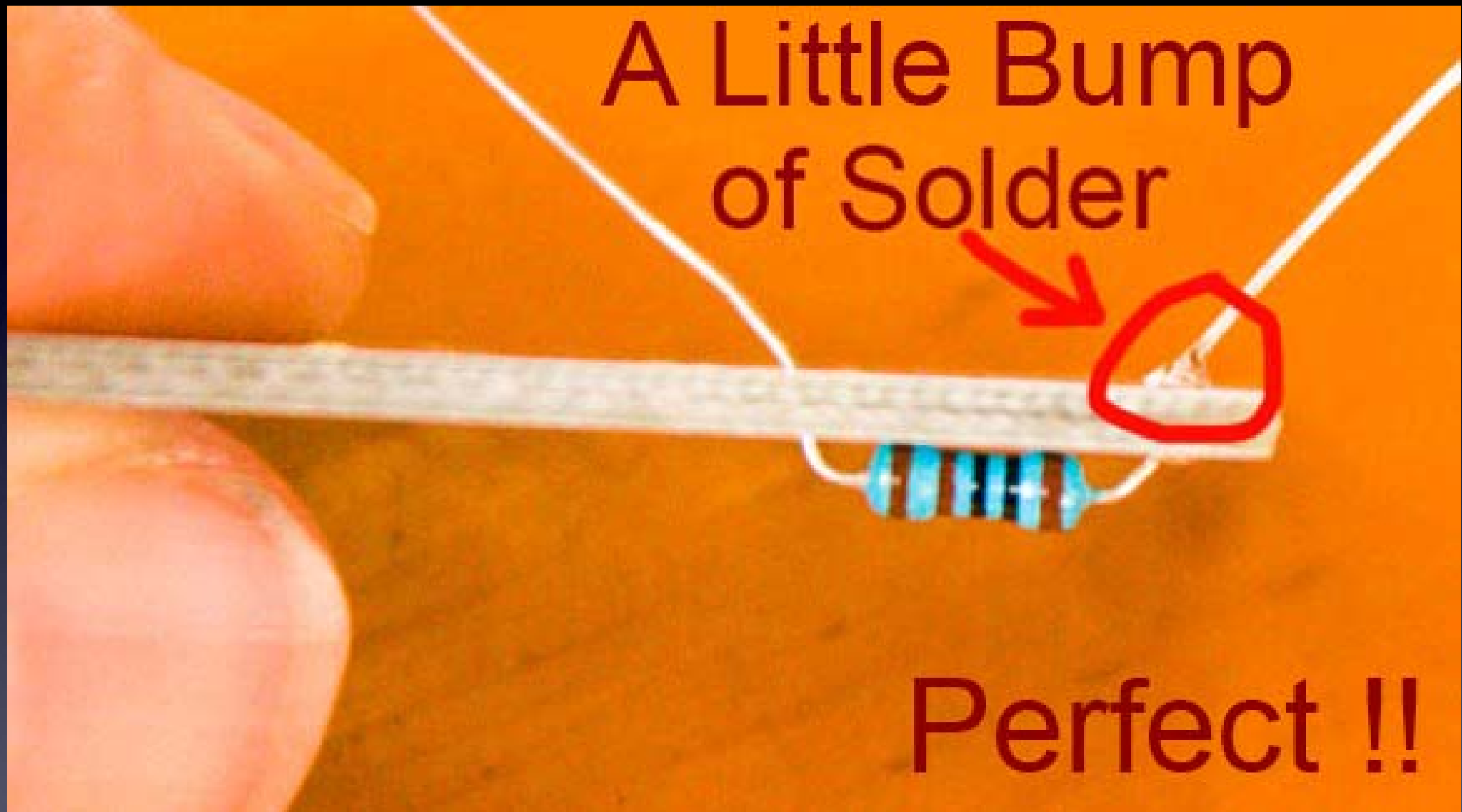
Keep hot tip down  
1 second  
for solder to flow !!



**Now**

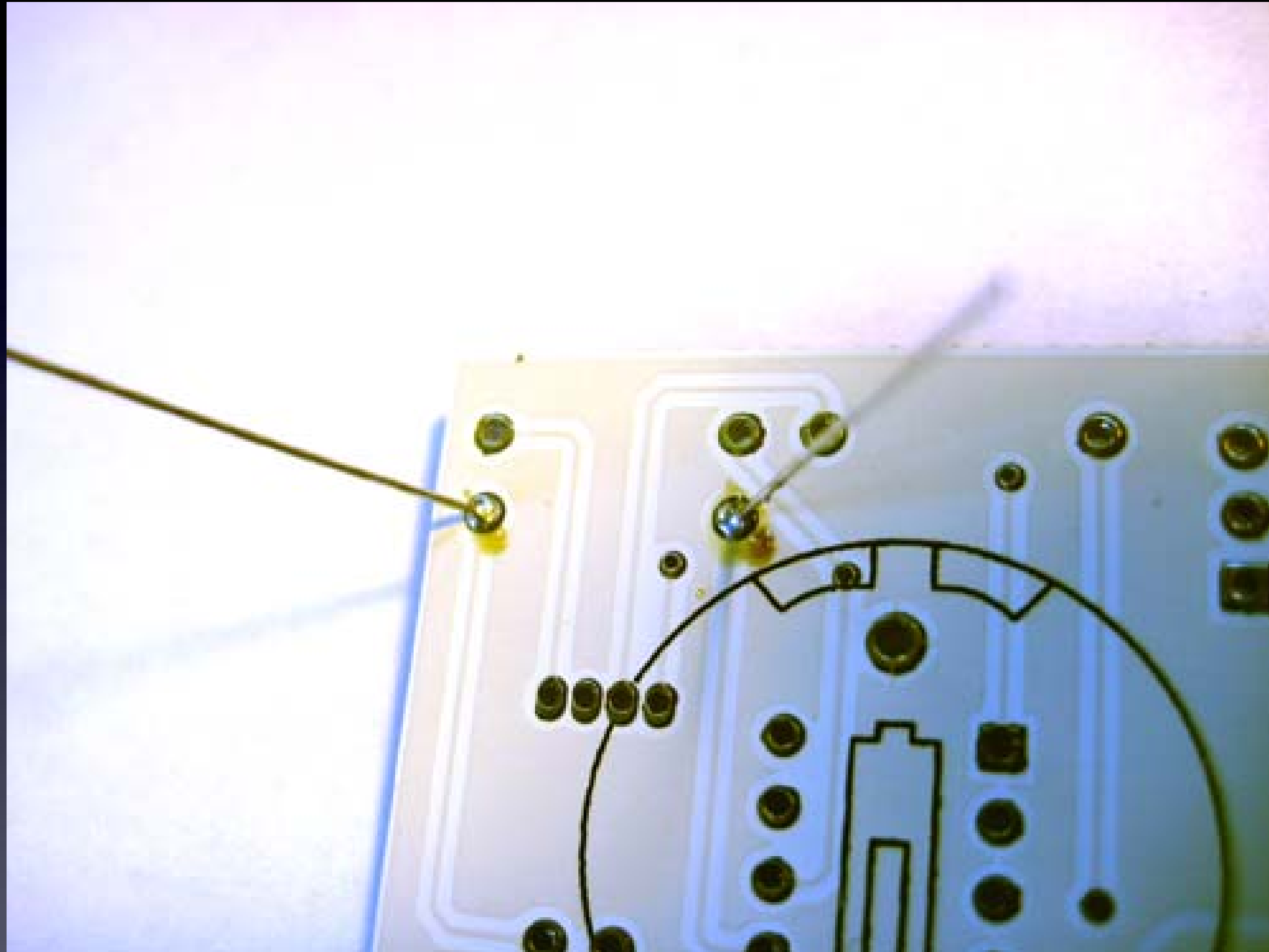
**Lift soldering iron**





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.

Solder all of the leads of the part to the board

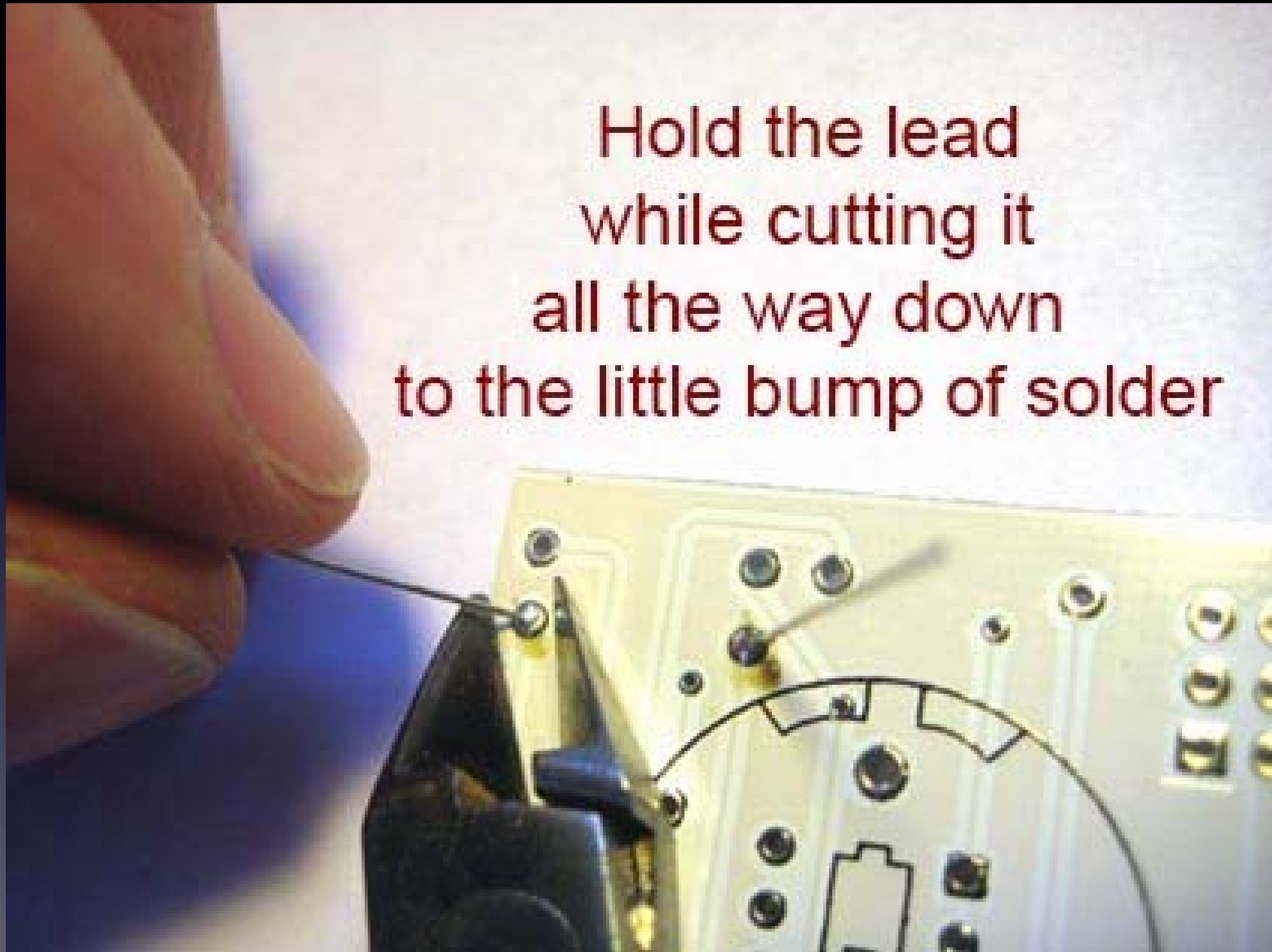


For this part, there are two leads

Here you can see two good solder connections

## Now cut the leads short

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



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

# Safety Tip #3:

Hold or cover the lead !

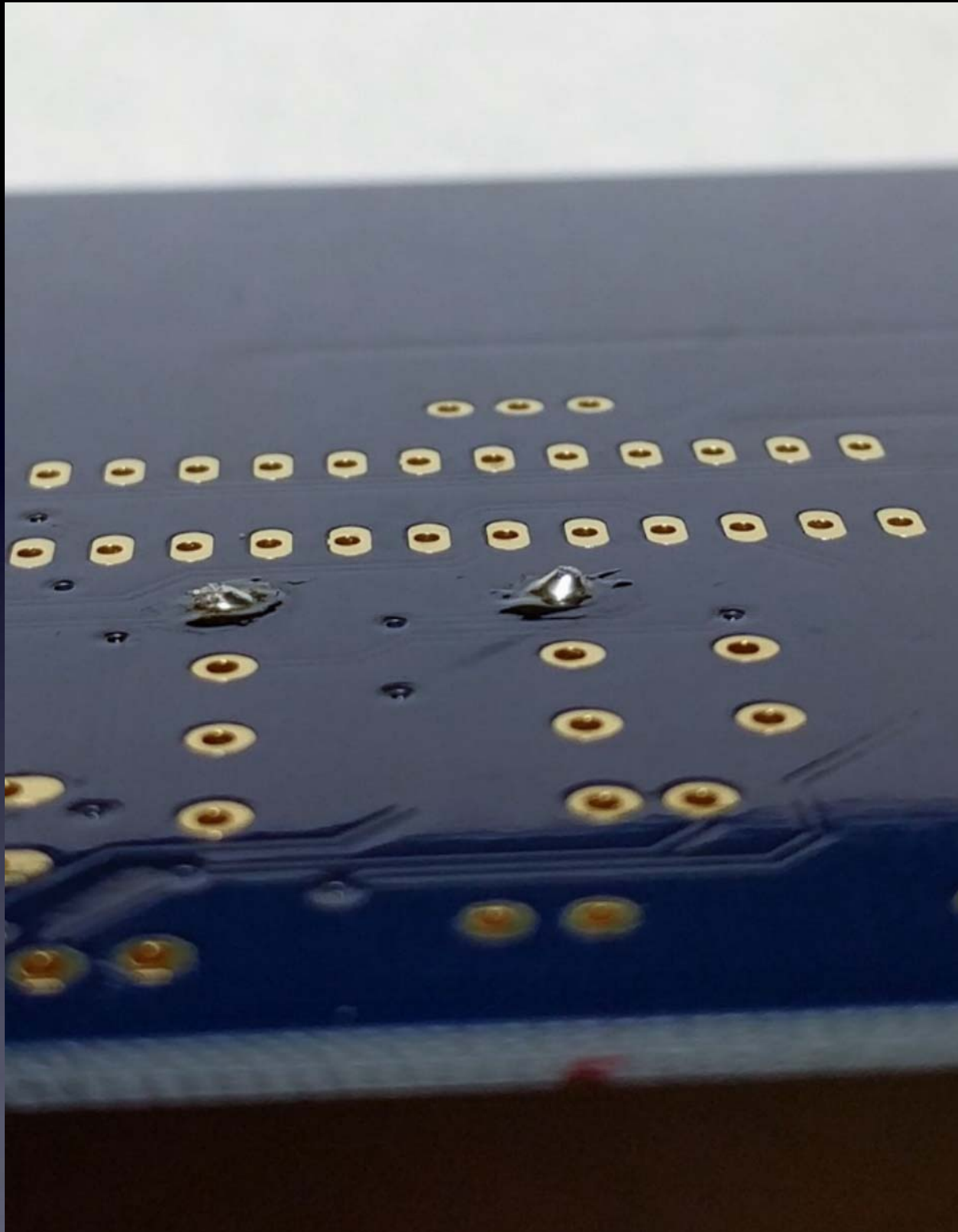
(or it will fly into your eye!)

*(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

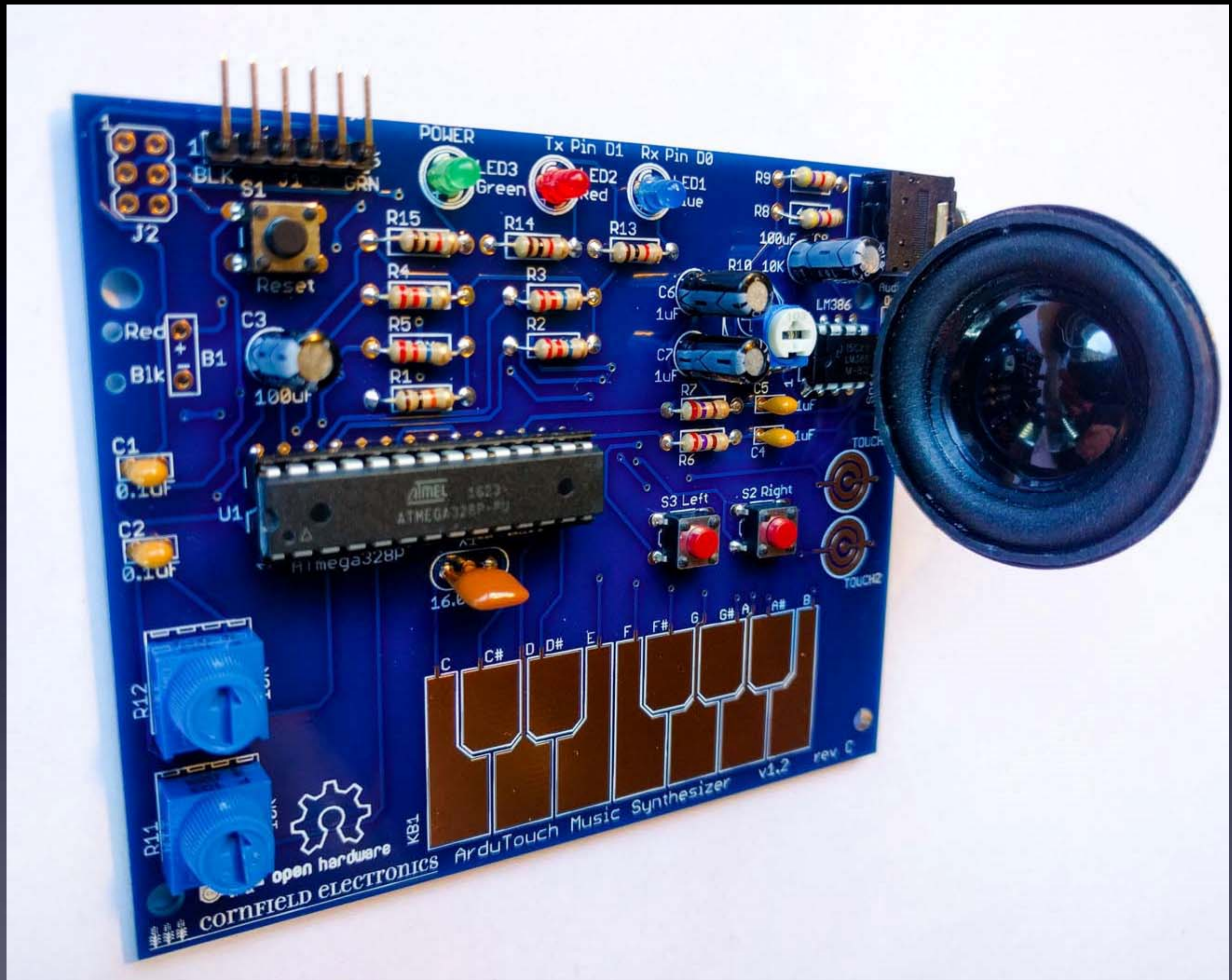
Notice that:

- each connection is a small bump (not flat)
- you cannot see any pad (it's totally covered with solder)
- you cannot see the hole (it's totally covered with solder)

One part at a time

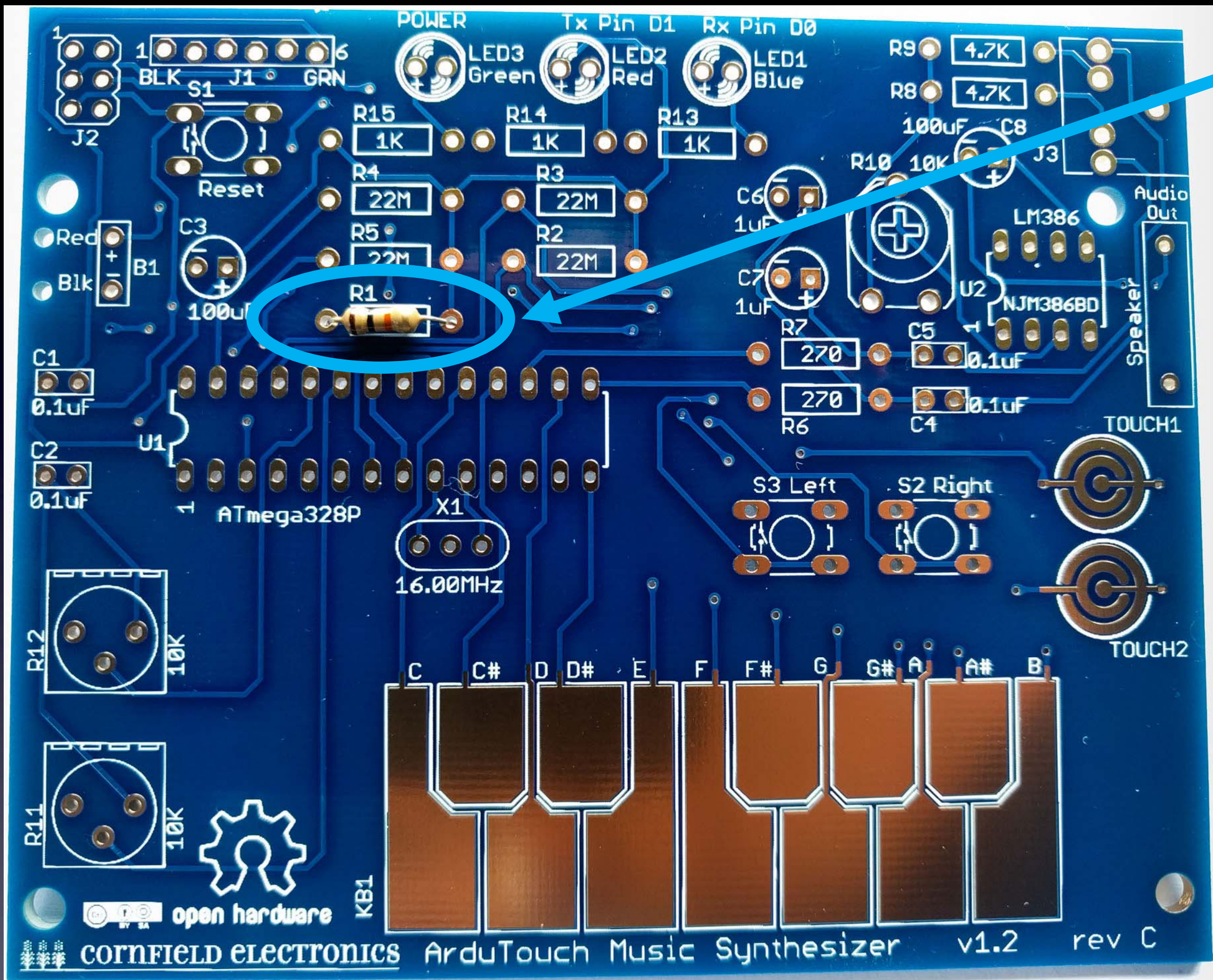


Till all the parts are soldered



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

Let's start!



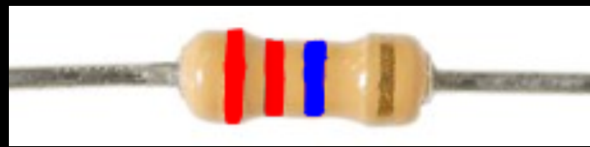
If you haven't done so already, solder R1: brown, black, orange

R1:



10K: Brown, Black, Orange

R2, R3, R4, R5:



22M: Red, Red, Blue

R6, R7:



270: Red, Violet, Brown

R8, R9:

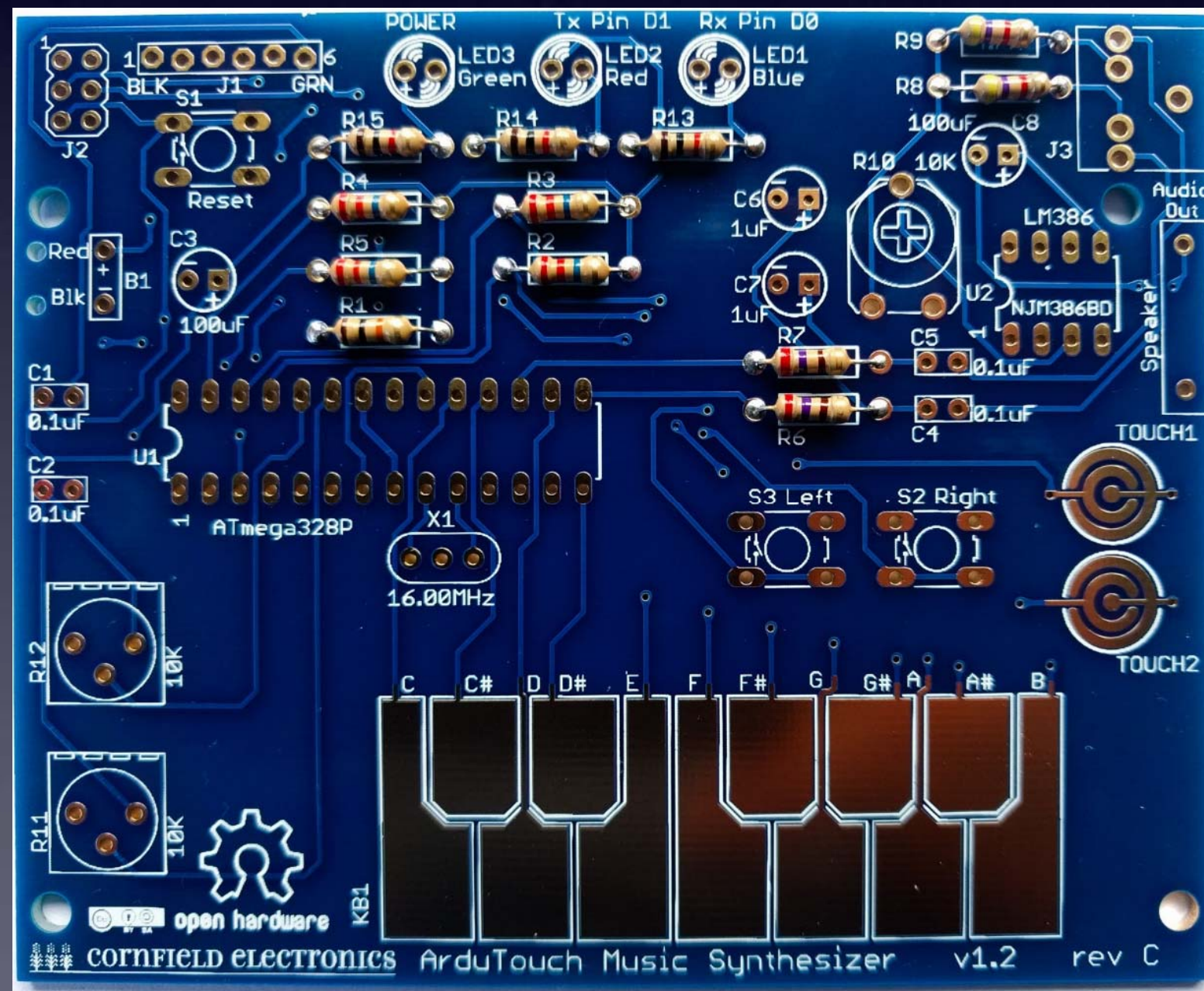


4.7K: Yellow, Violet, Red

R13, R14, R15:

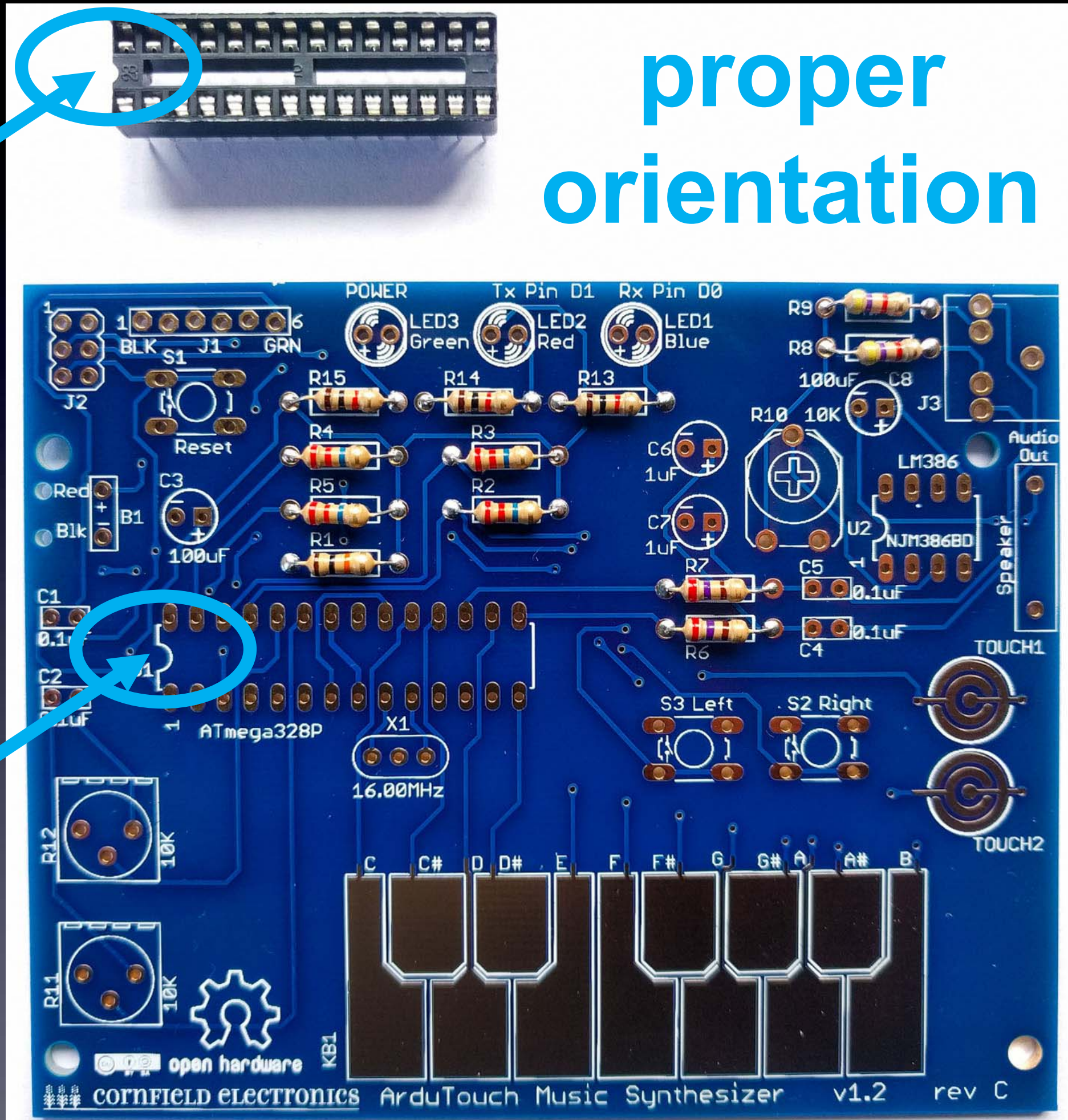


1K: Brown, Black, Red

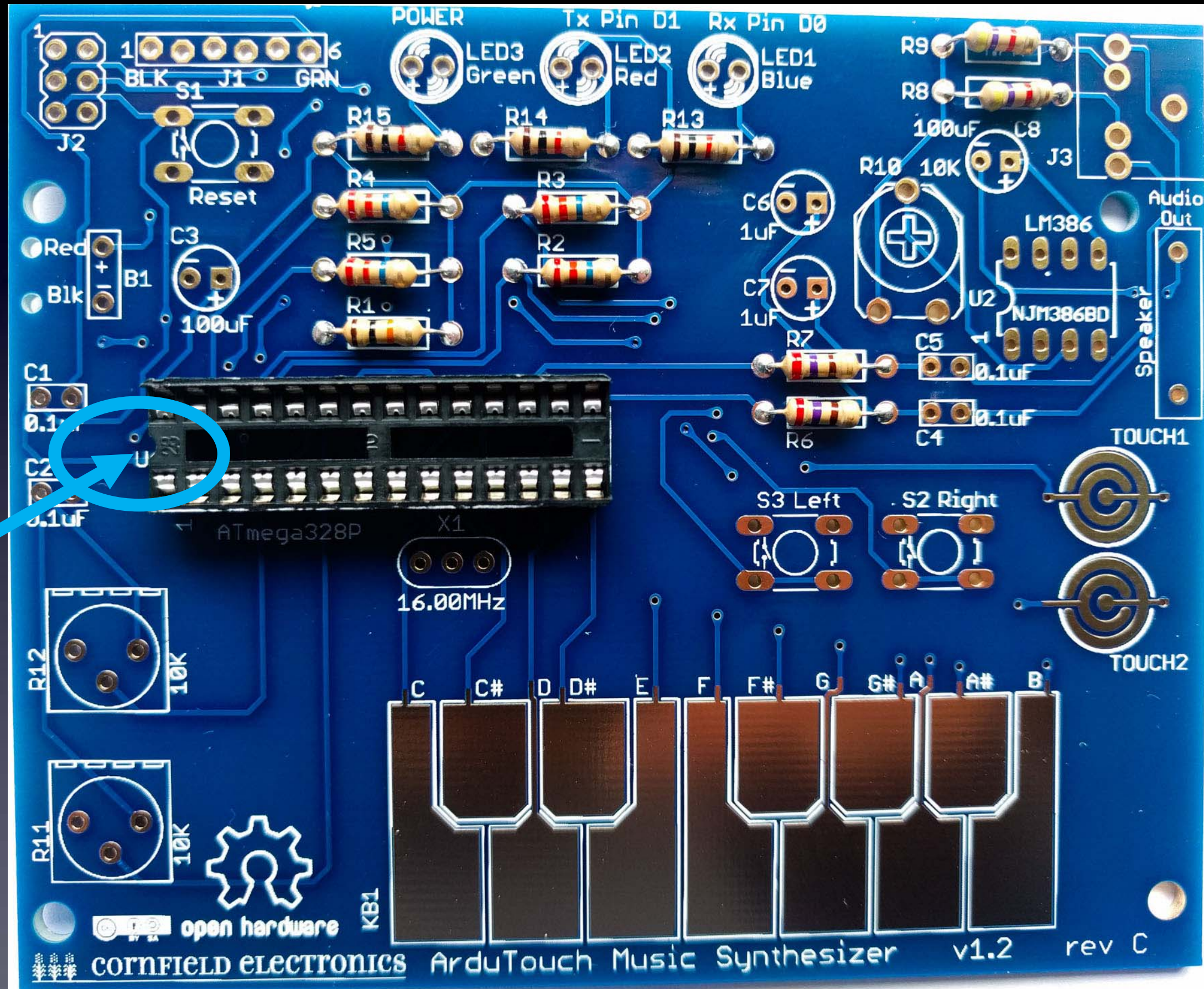


# U1: microcontroller socket

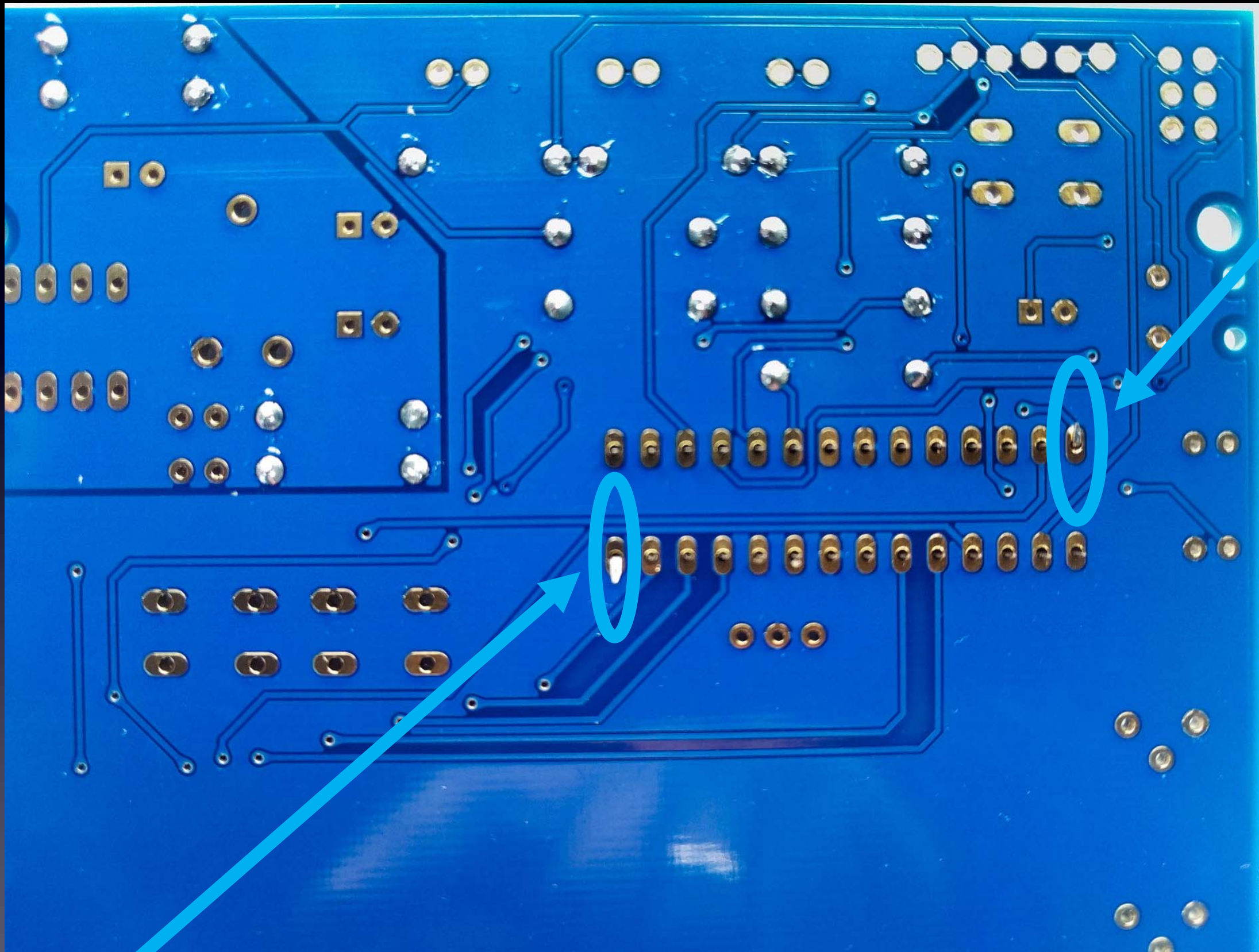
proper orientation



# U1: microcontroller socket: inserted correctly



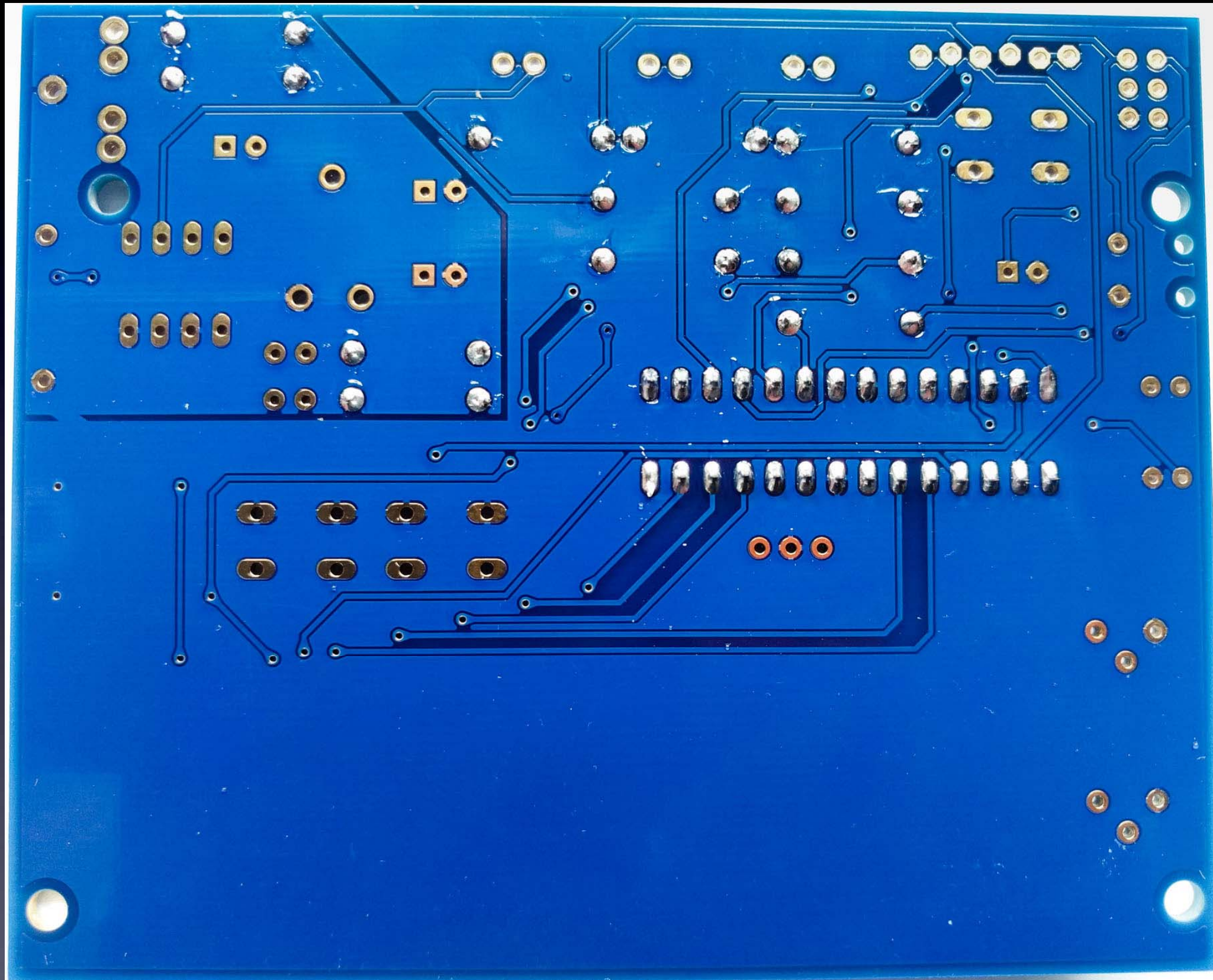
# U1: microcontroller socket



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

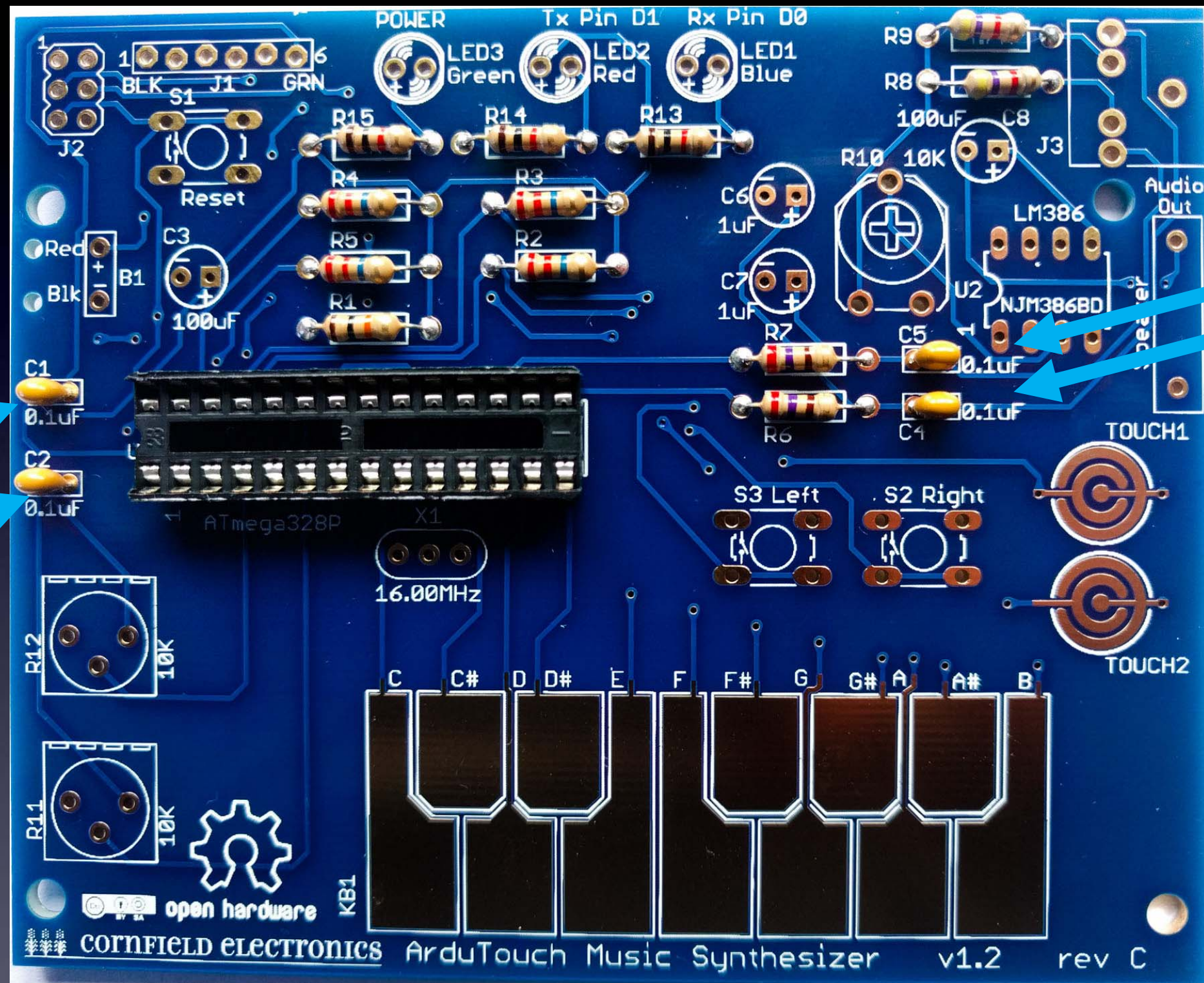


# U1: microcontroller socket

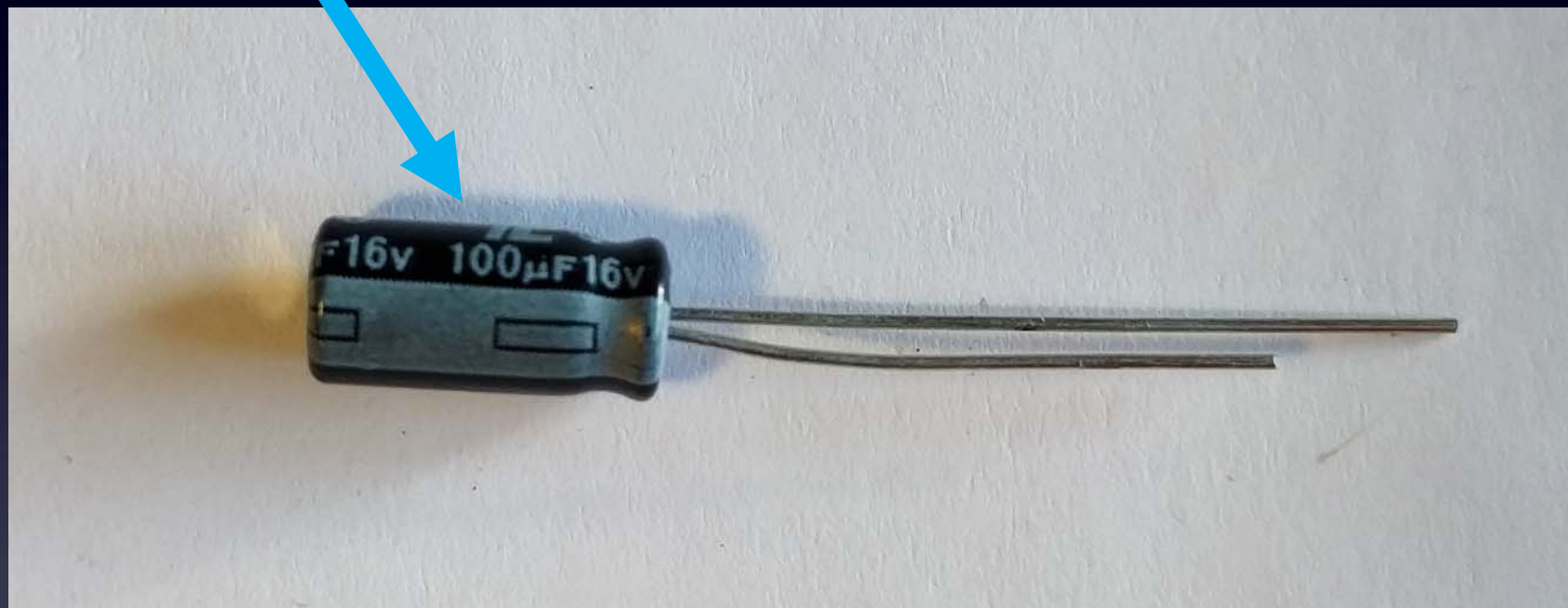


All 28 leads soldered to the board:

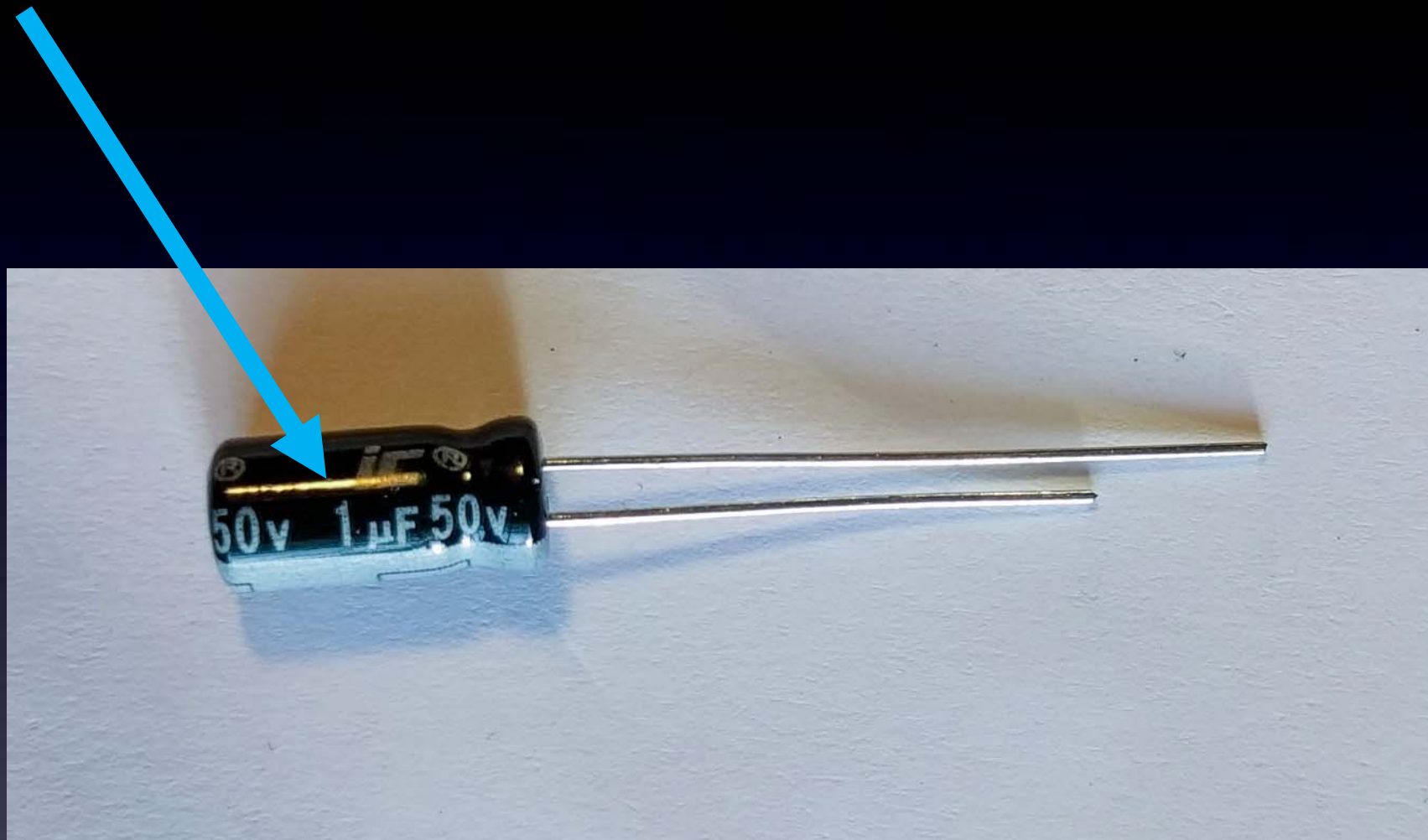
→ Notice that each has a little bump of solder (not flat). ←



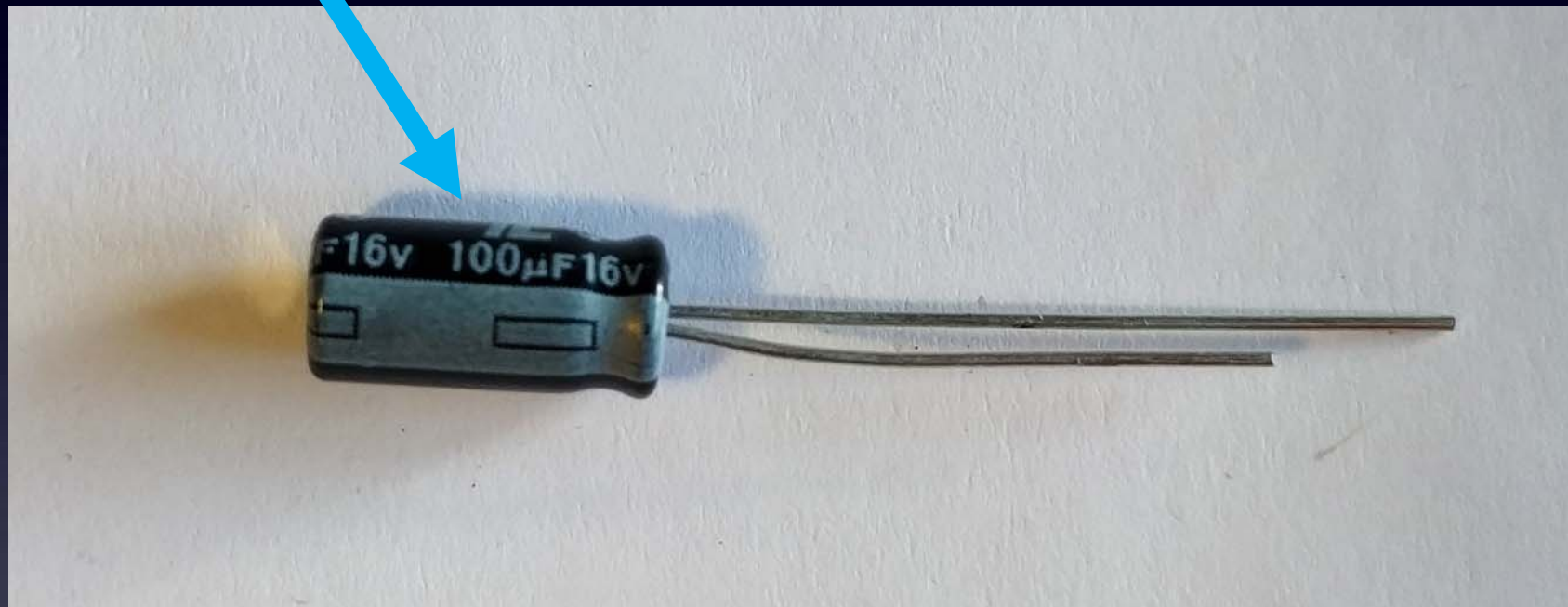
C1, C2, C4, C5



**C3, C8: 100uF**



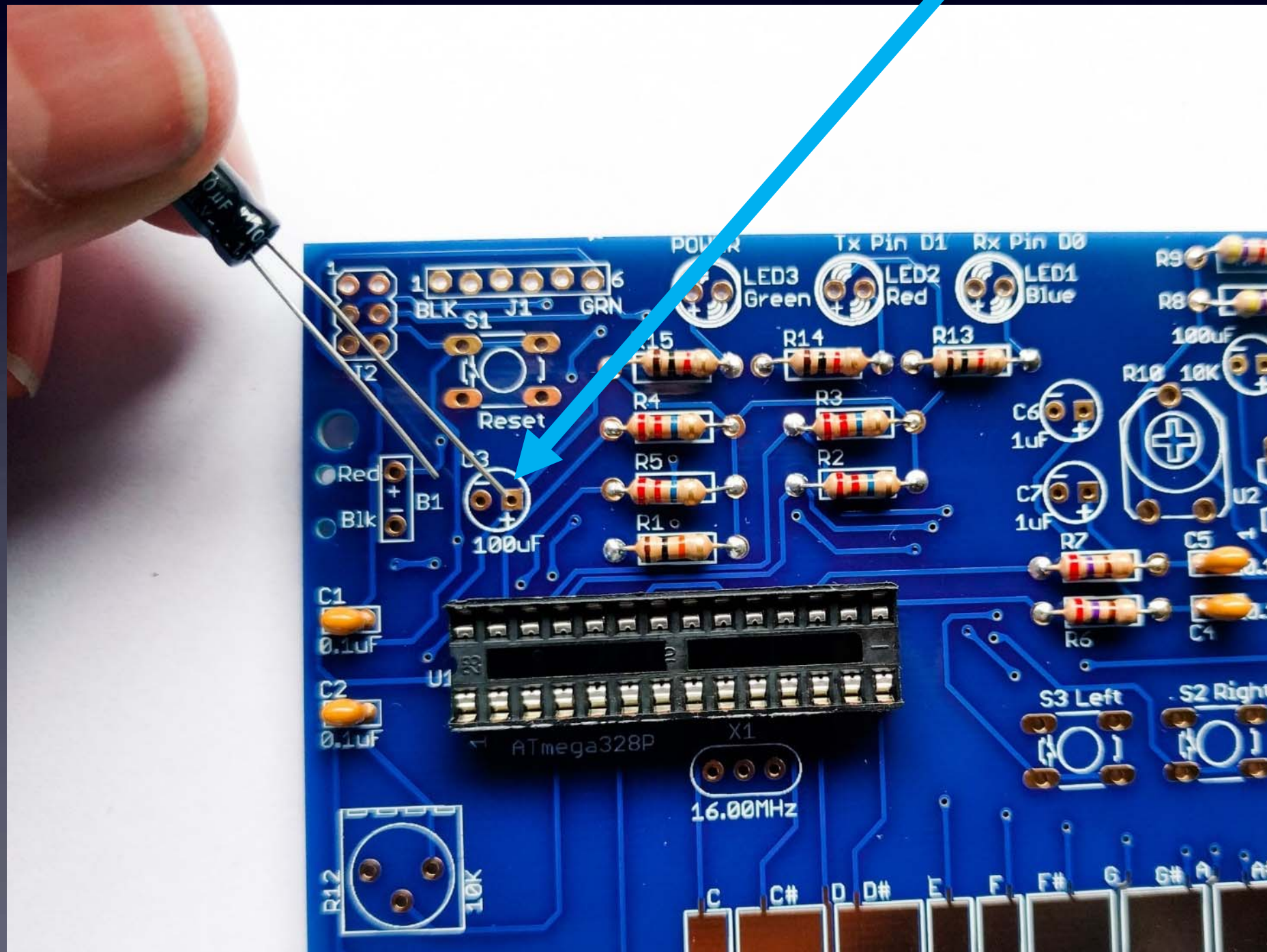
**Different than C3, C8!**  
**C6, C7: 1uF**

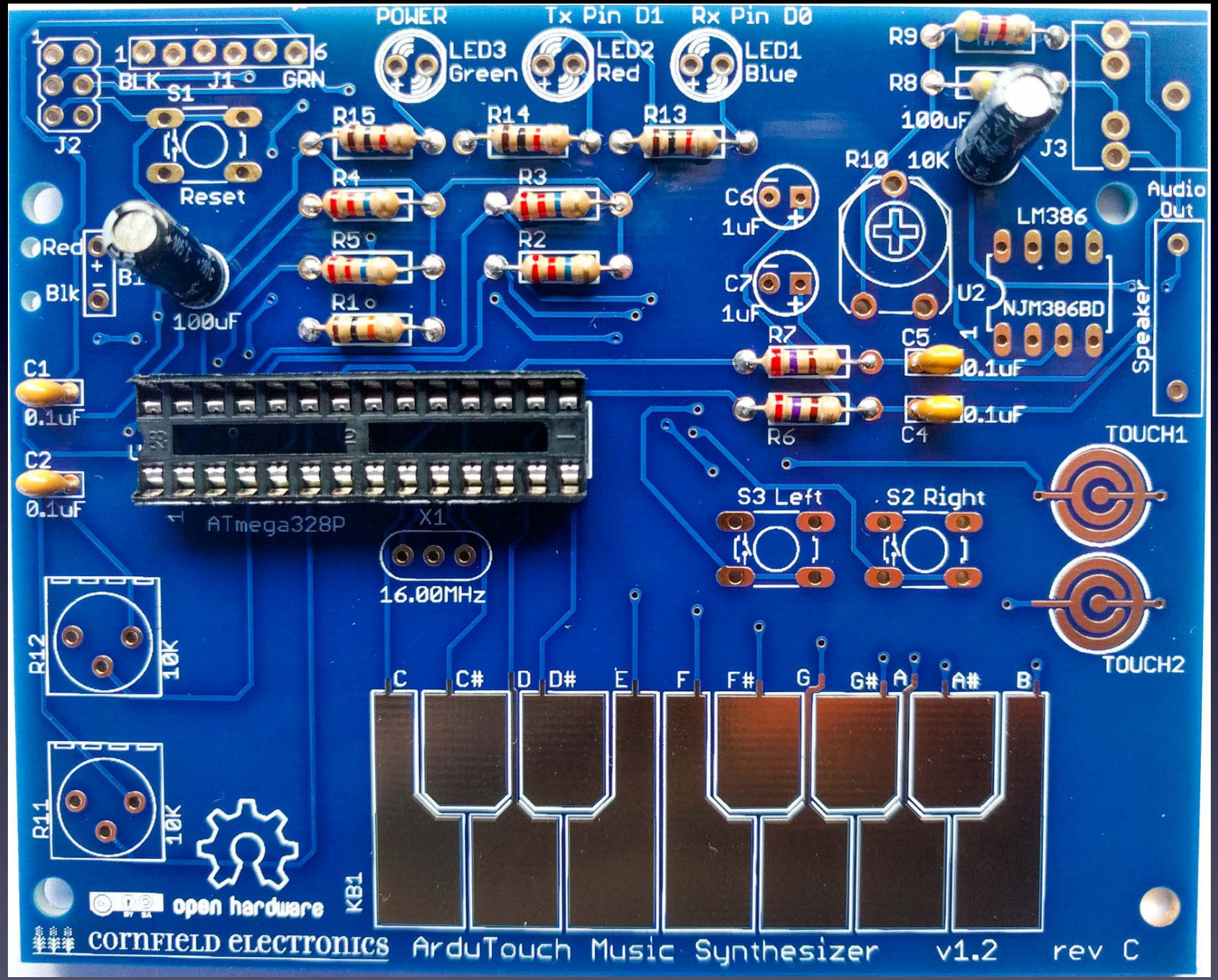


**C3, C8: 100uF**

**C3, C8:  
Long Lead “+”**

**Use 100uF !!**



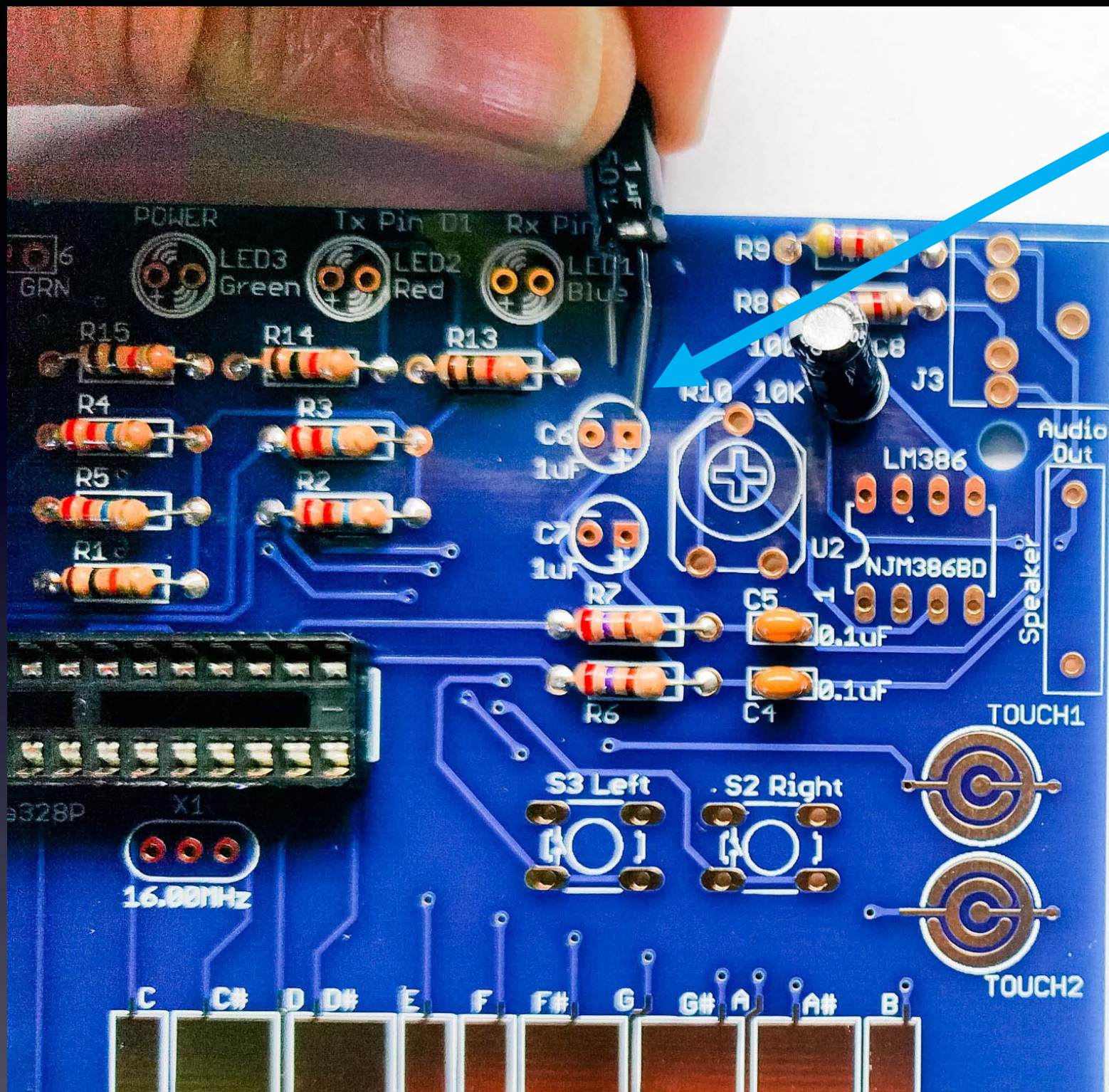


**C3, C8: 100uF – soldered to board**



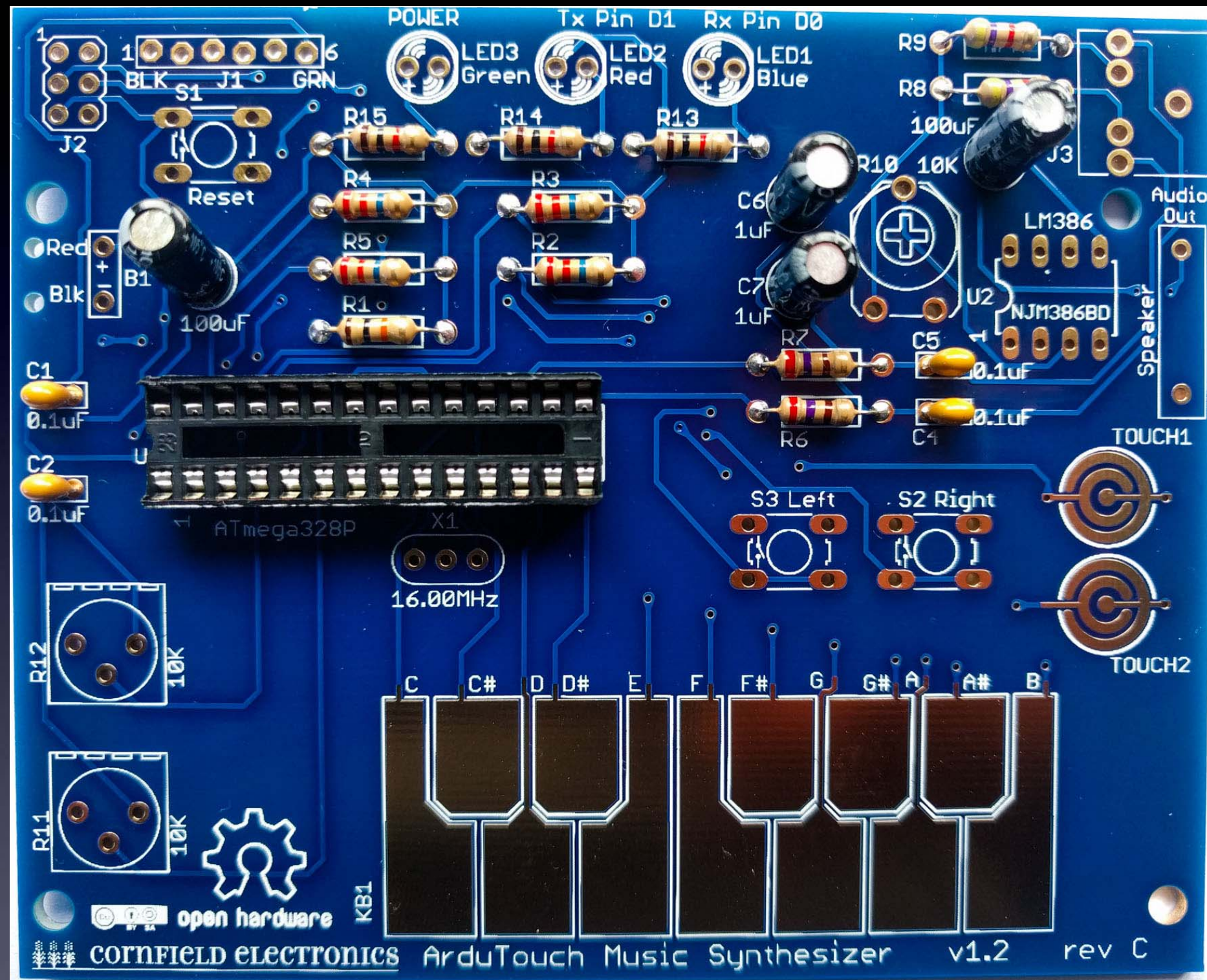
C6, C7: 1µF





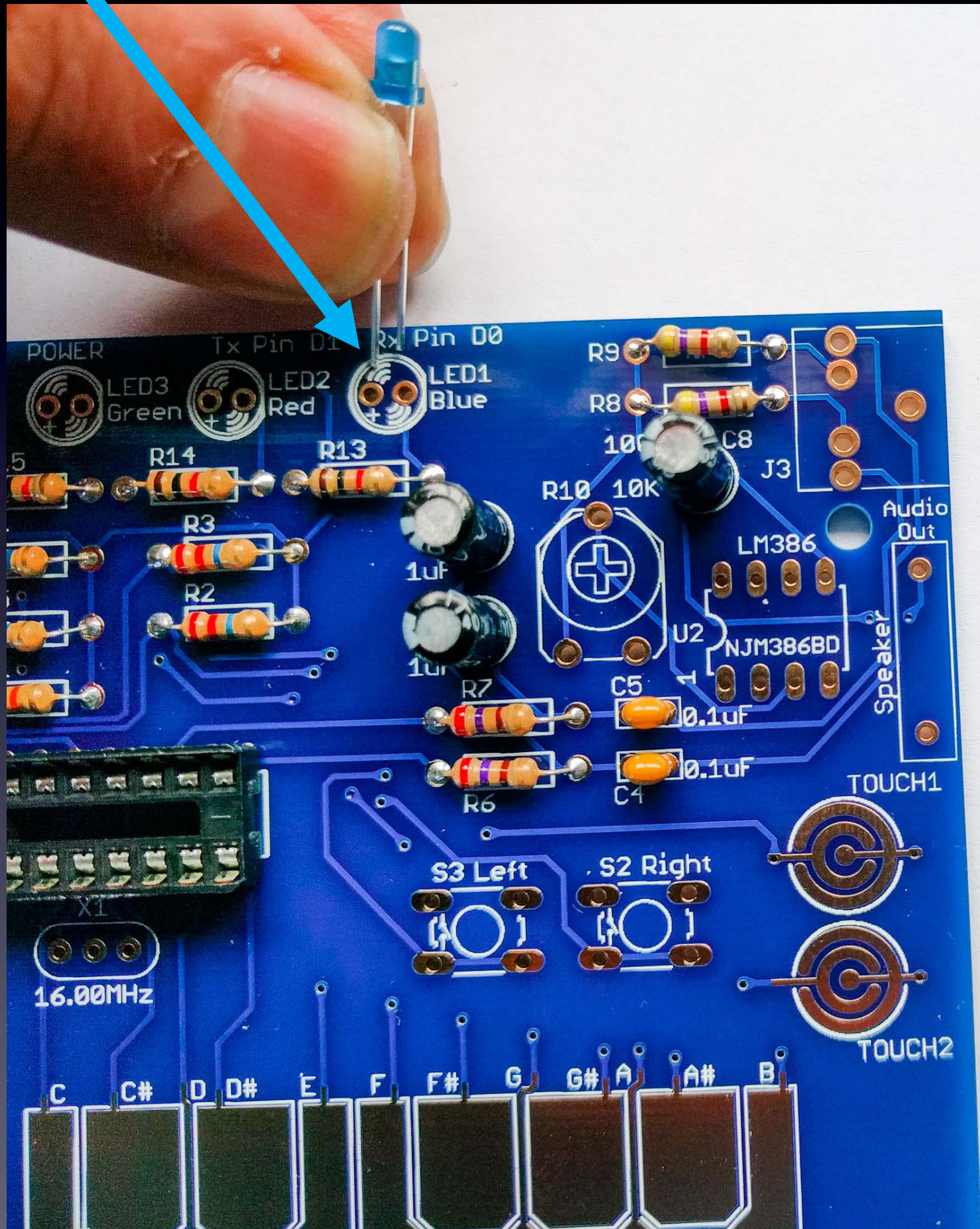
**C6, C7:  
Long Lead “+”**

**Use 1uF !!**



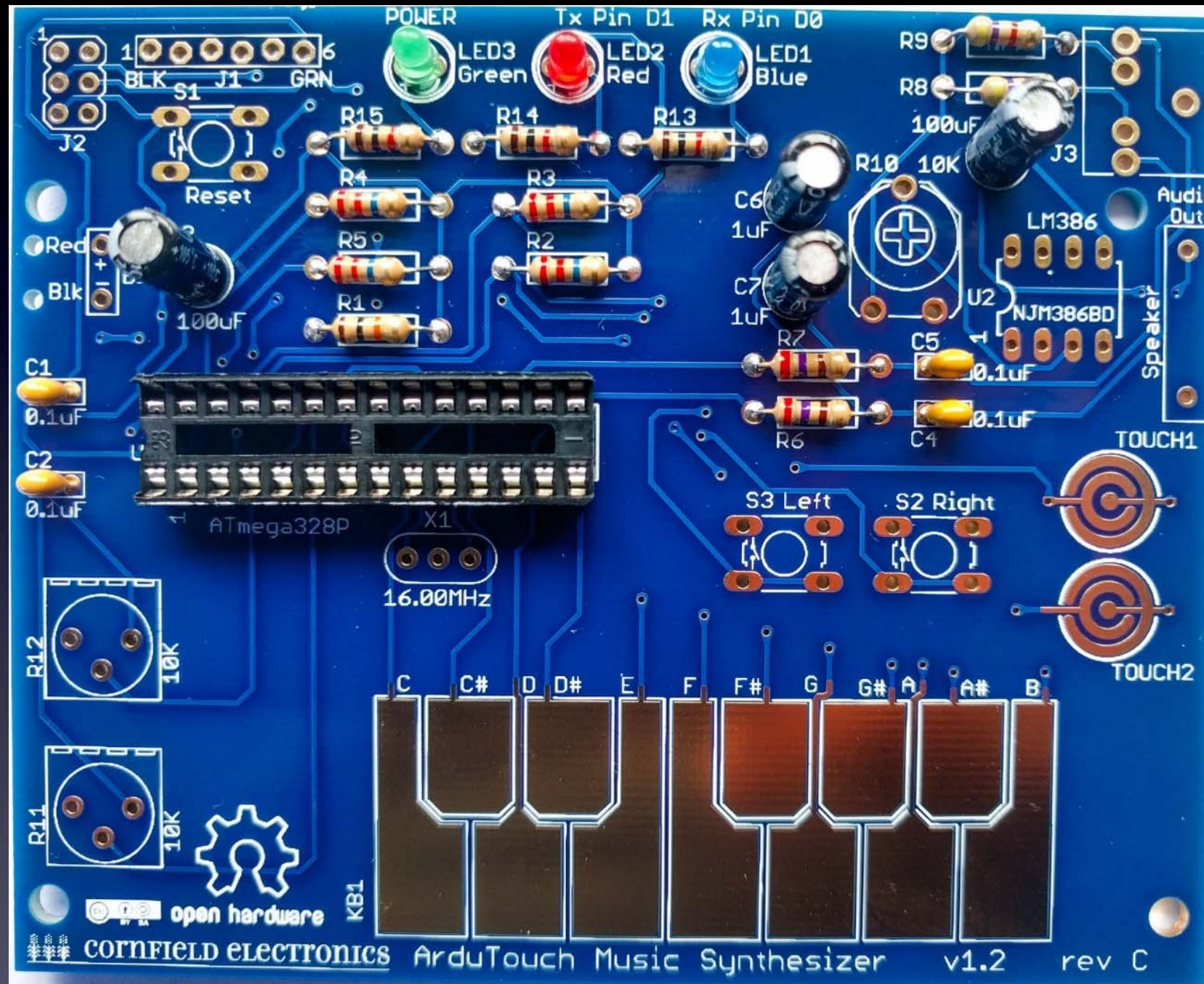
C6, C7: 1uF – soldered to board

# LED1, LED2, LED3: Long Lead “+”



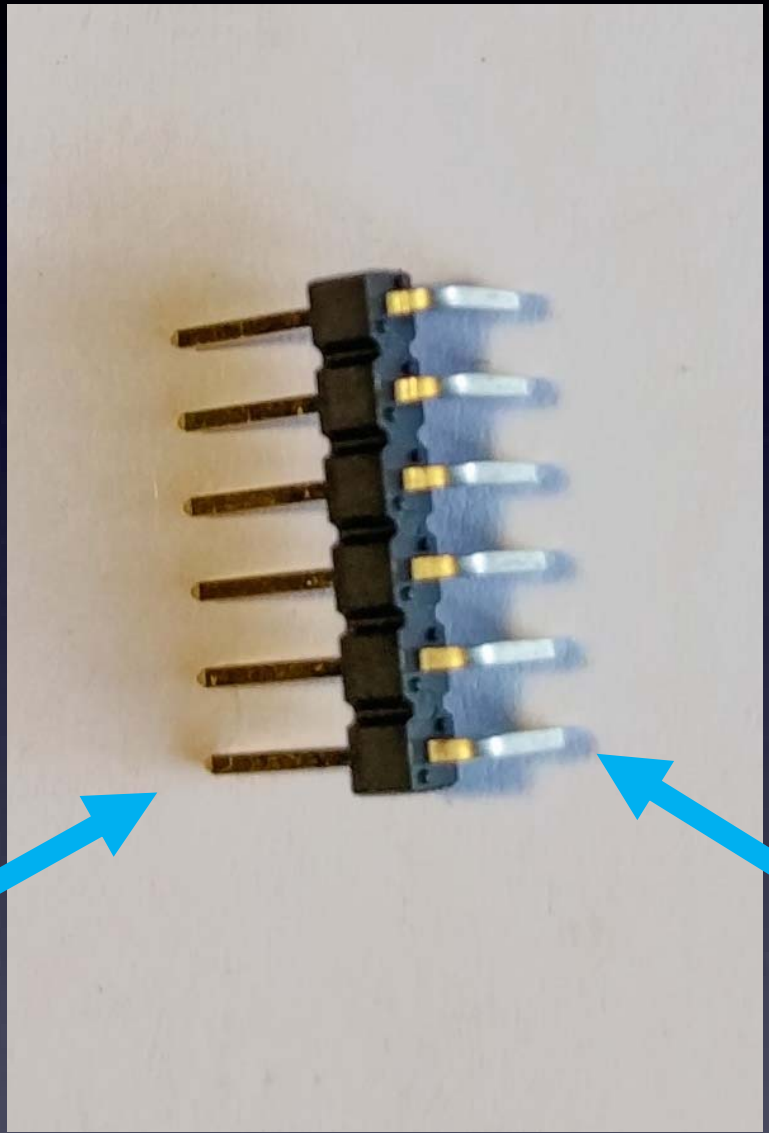
Save  
these leads

We'll use them for the speaker



**LED1, LED2, LED3**

**Green, Red, Blue – soldered to board**

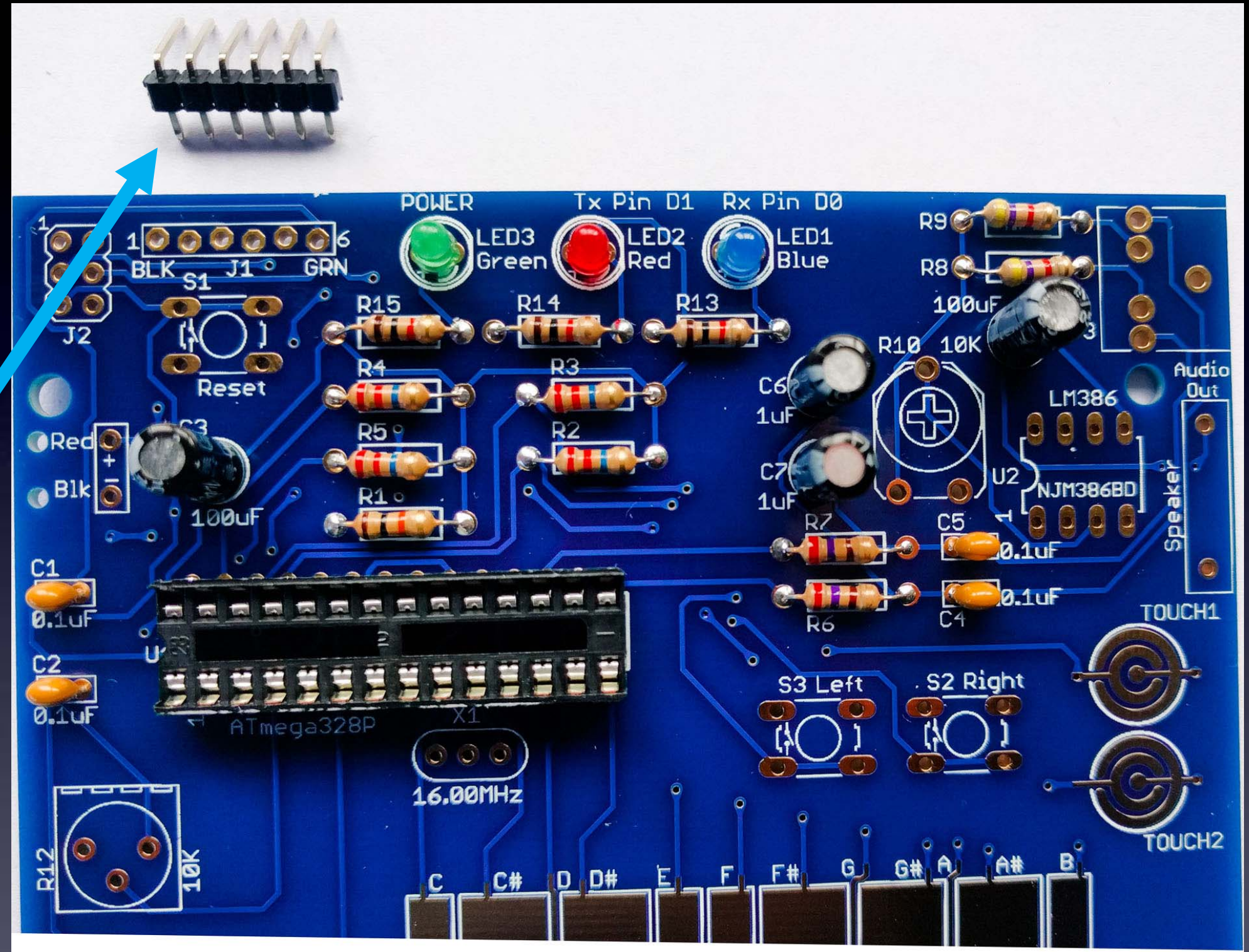


long leads

short leads

**J1**

# Short leads into board

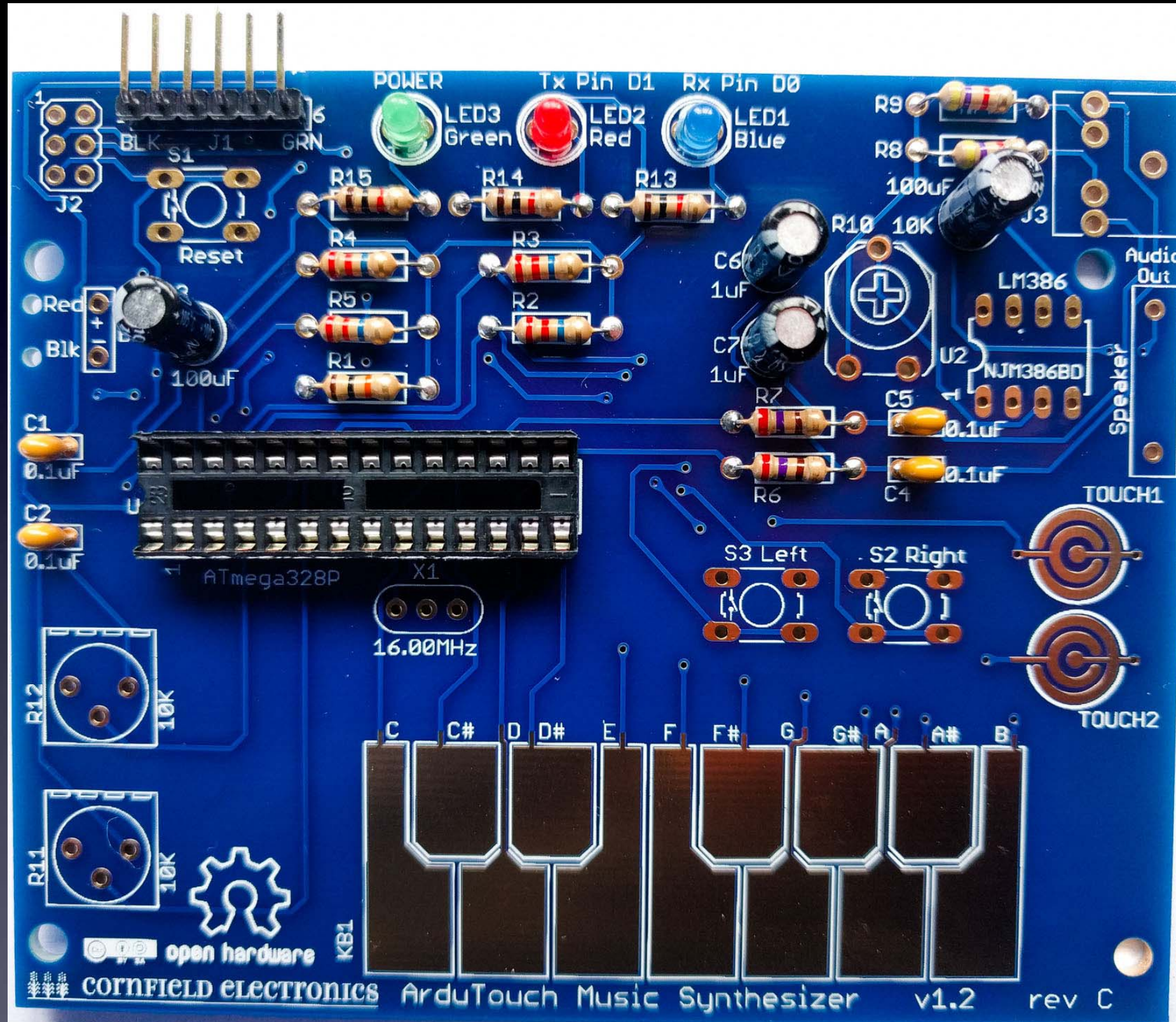


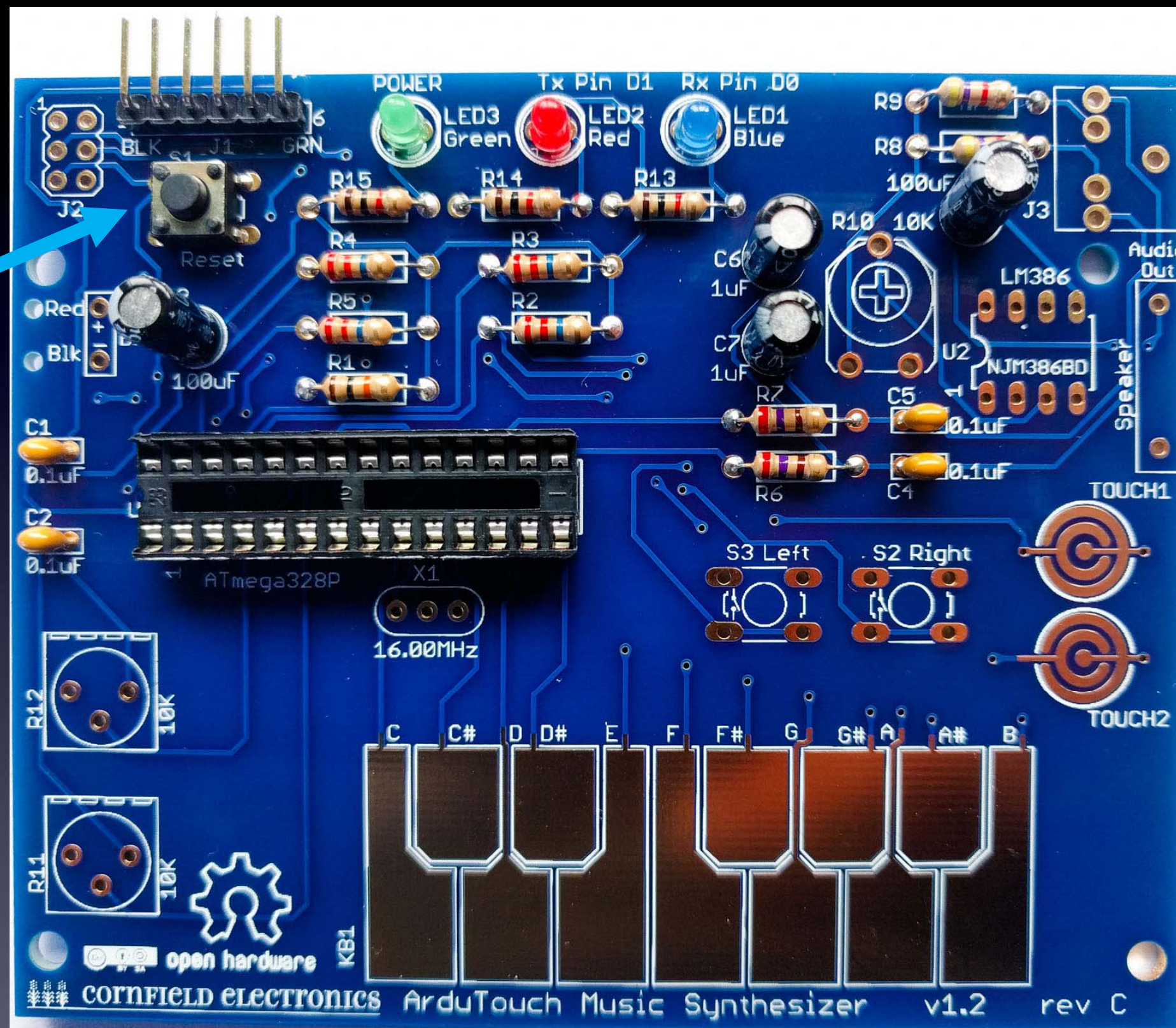
J1

short leads  
go into the board

→ long leads sticking out from  
board

J1





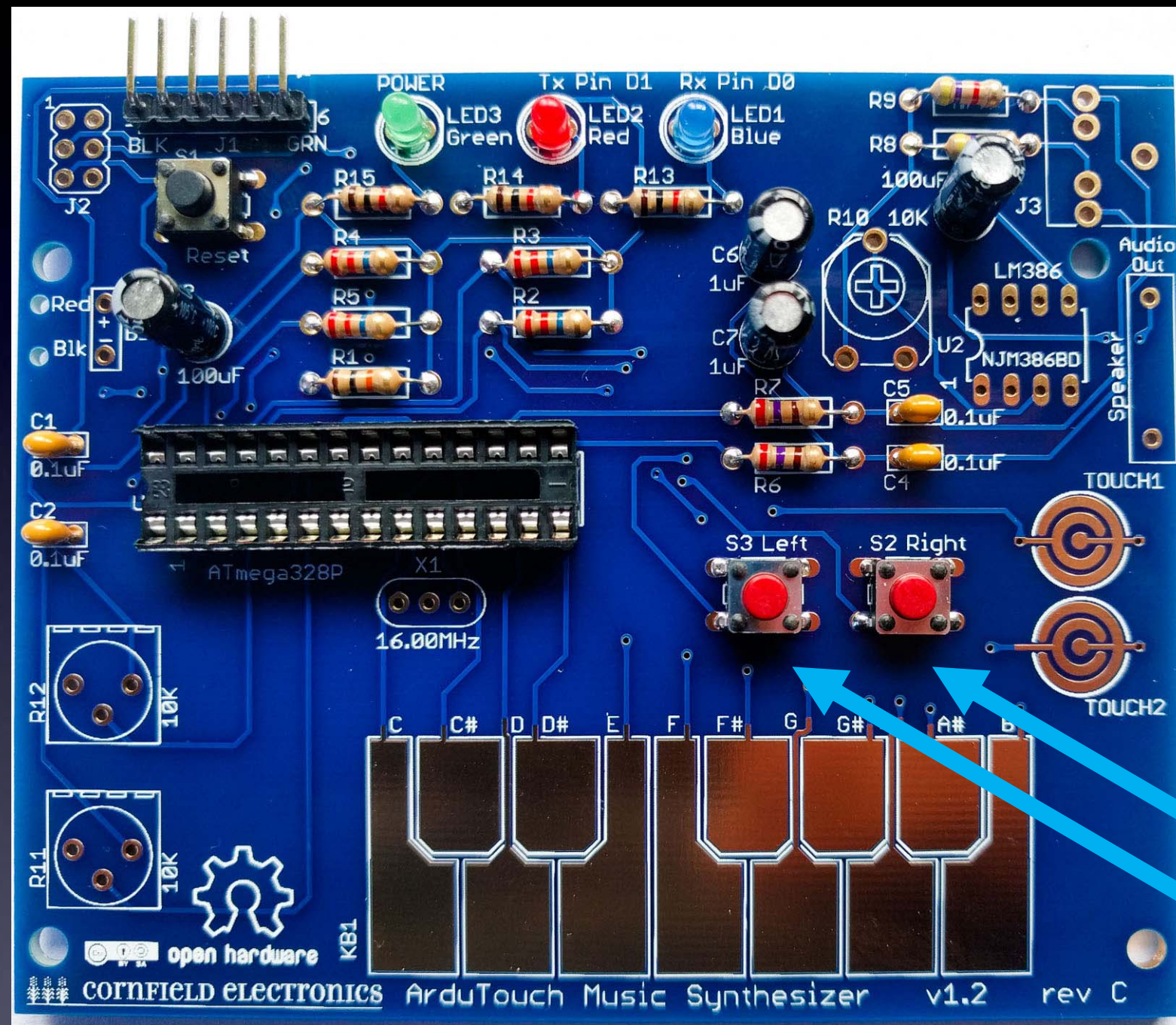
# S1: black Reset button

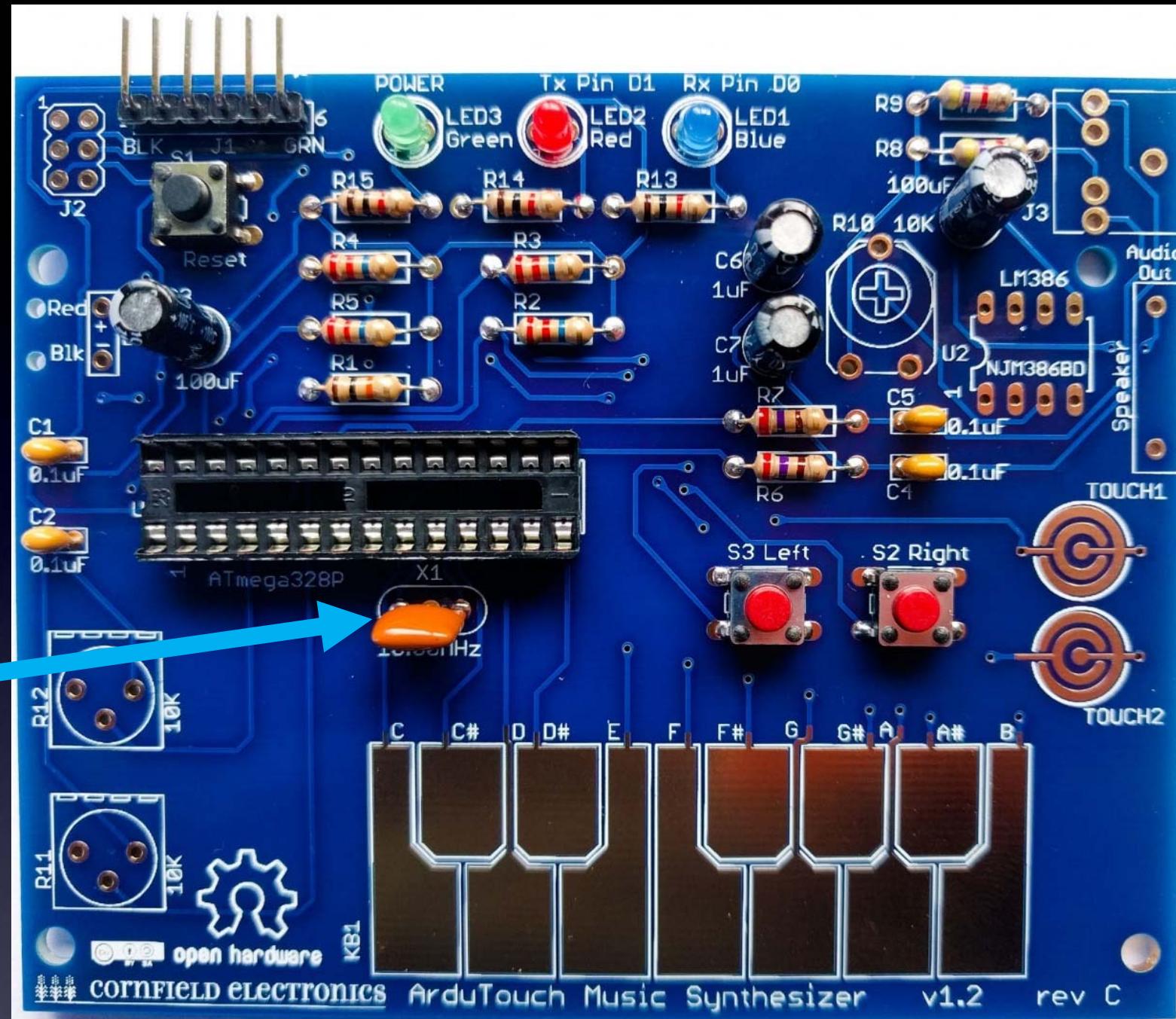
Note: The color of this switch is not important (some kits may have different colors).



# S2, S3: Red buttons

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



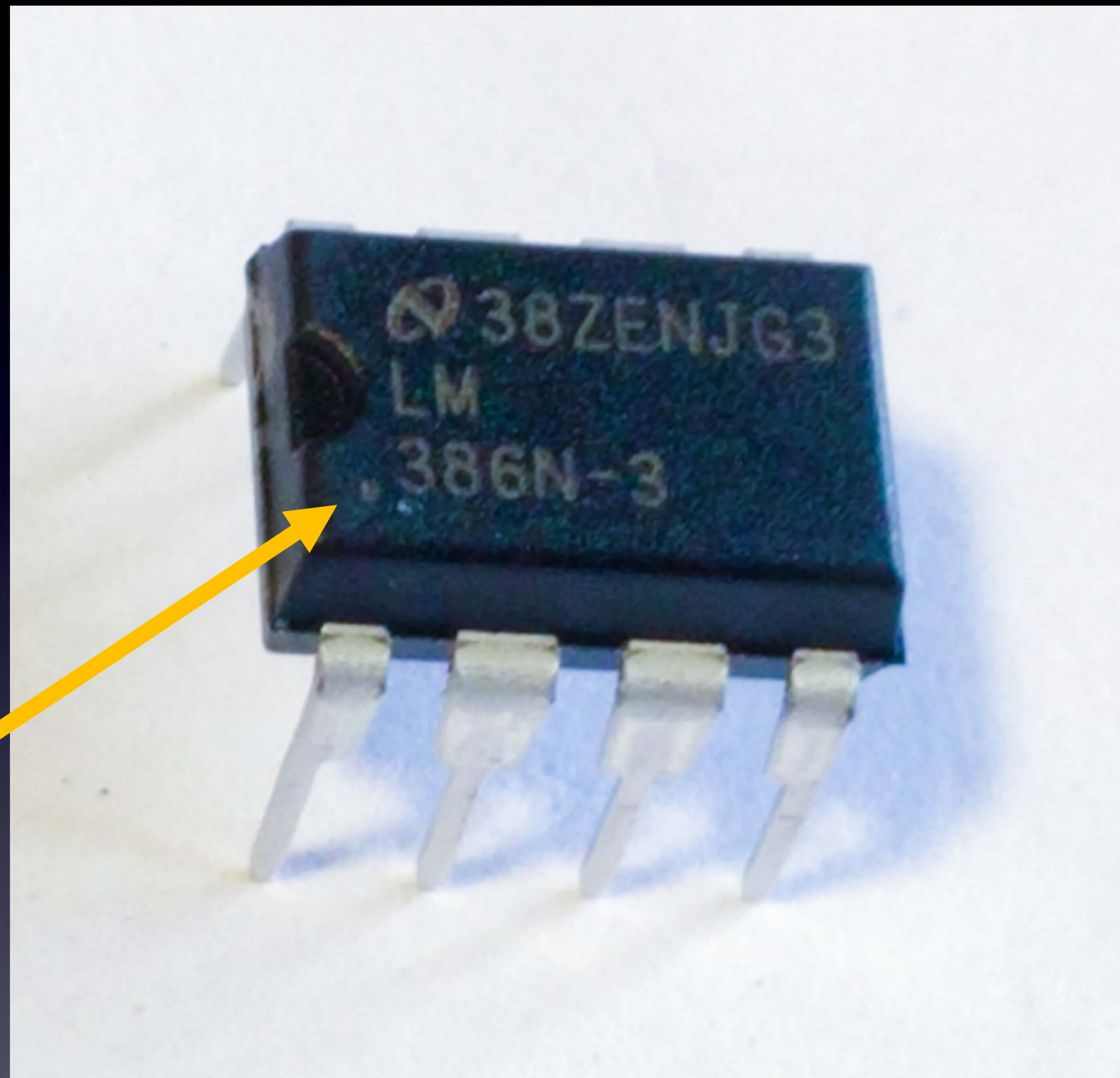


**X1**

**The orientation of X1 does not matter.**

Note: X1 may be yellow or blue.

**U2**

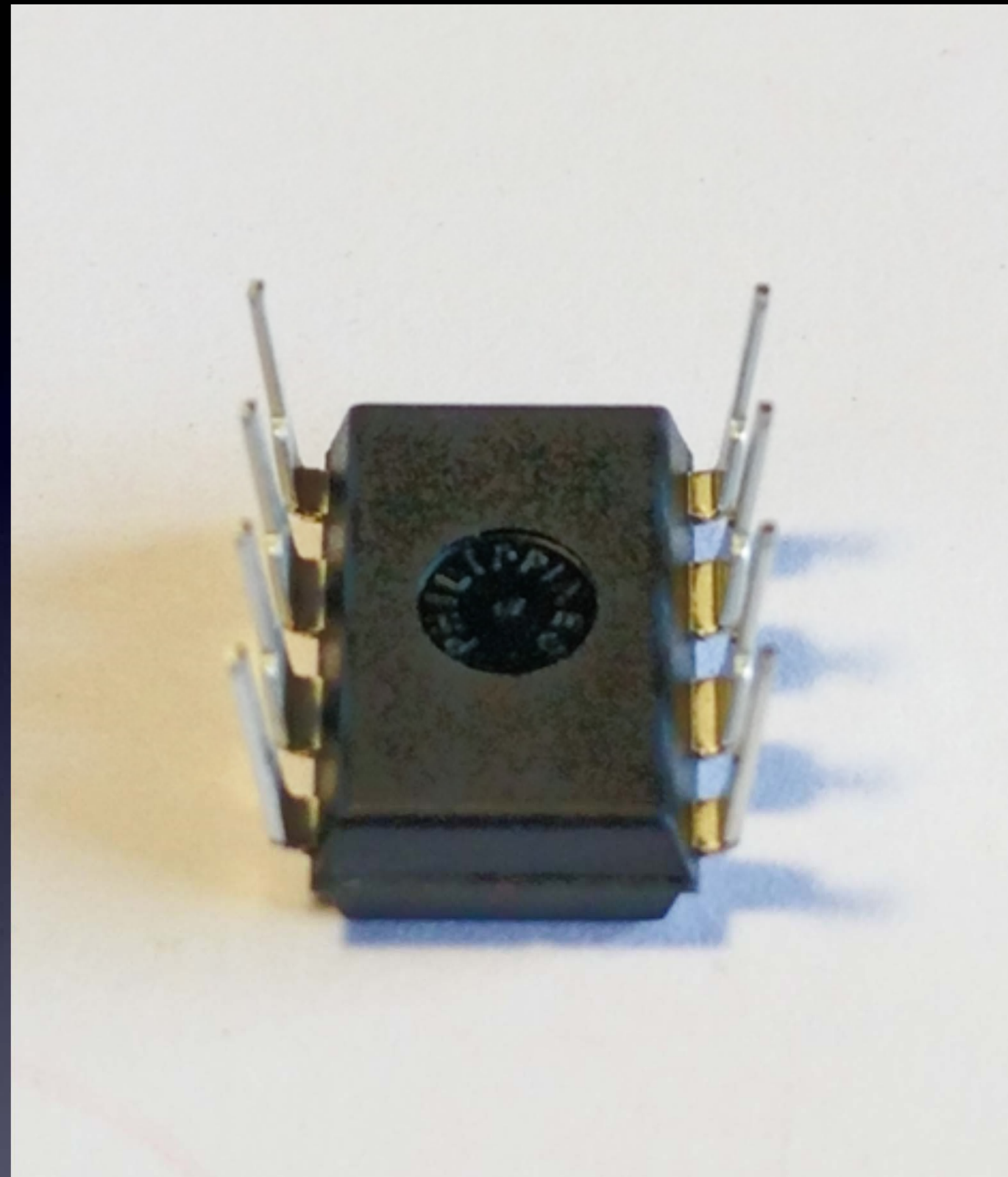


Indented black dot  
Pin 1

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

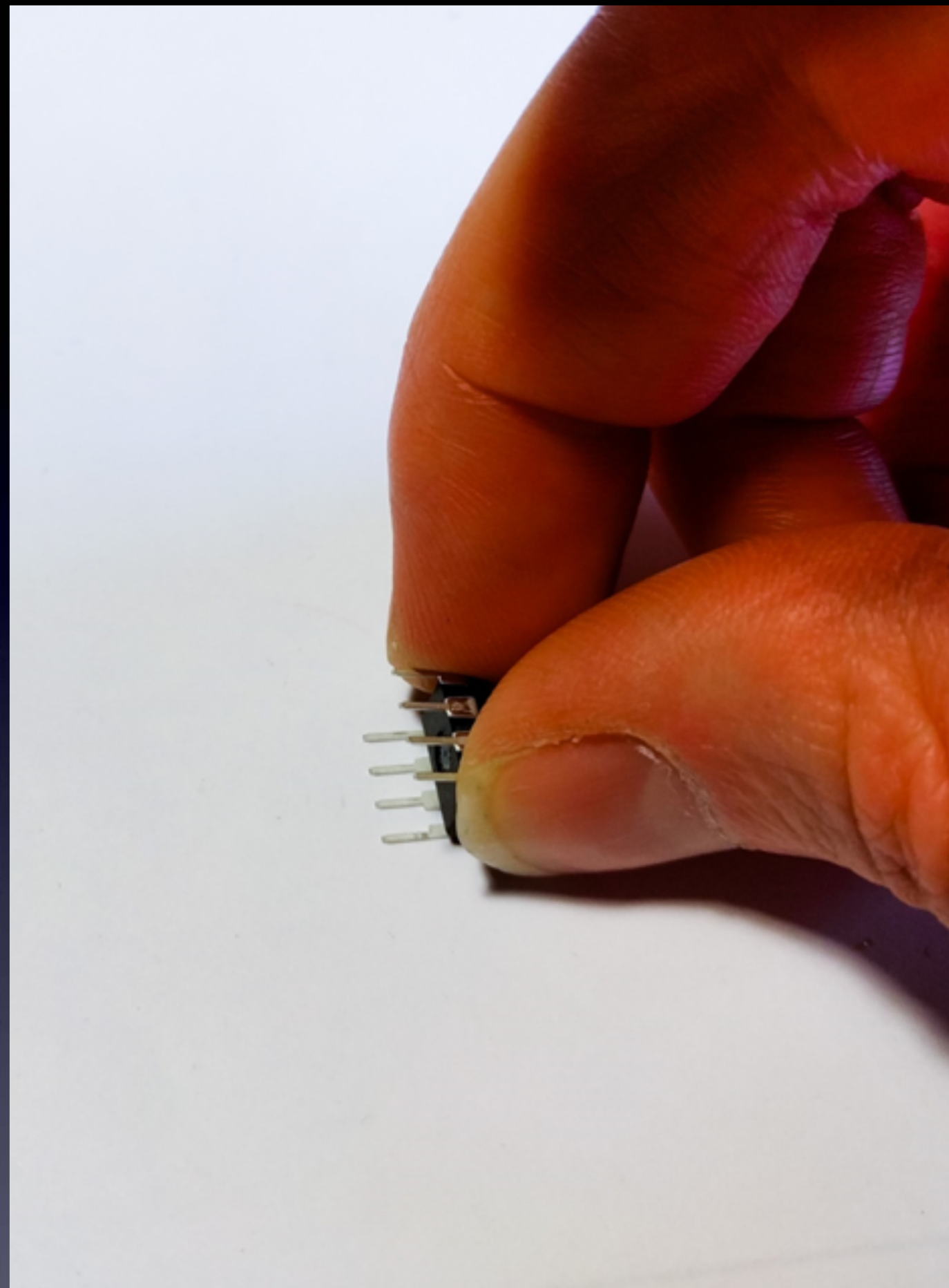
Note: Your chip may or may not have the indented half-moon at the left,  
it may have a black indented dot at the lower-left corner showing Pin 1.

**U2**



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

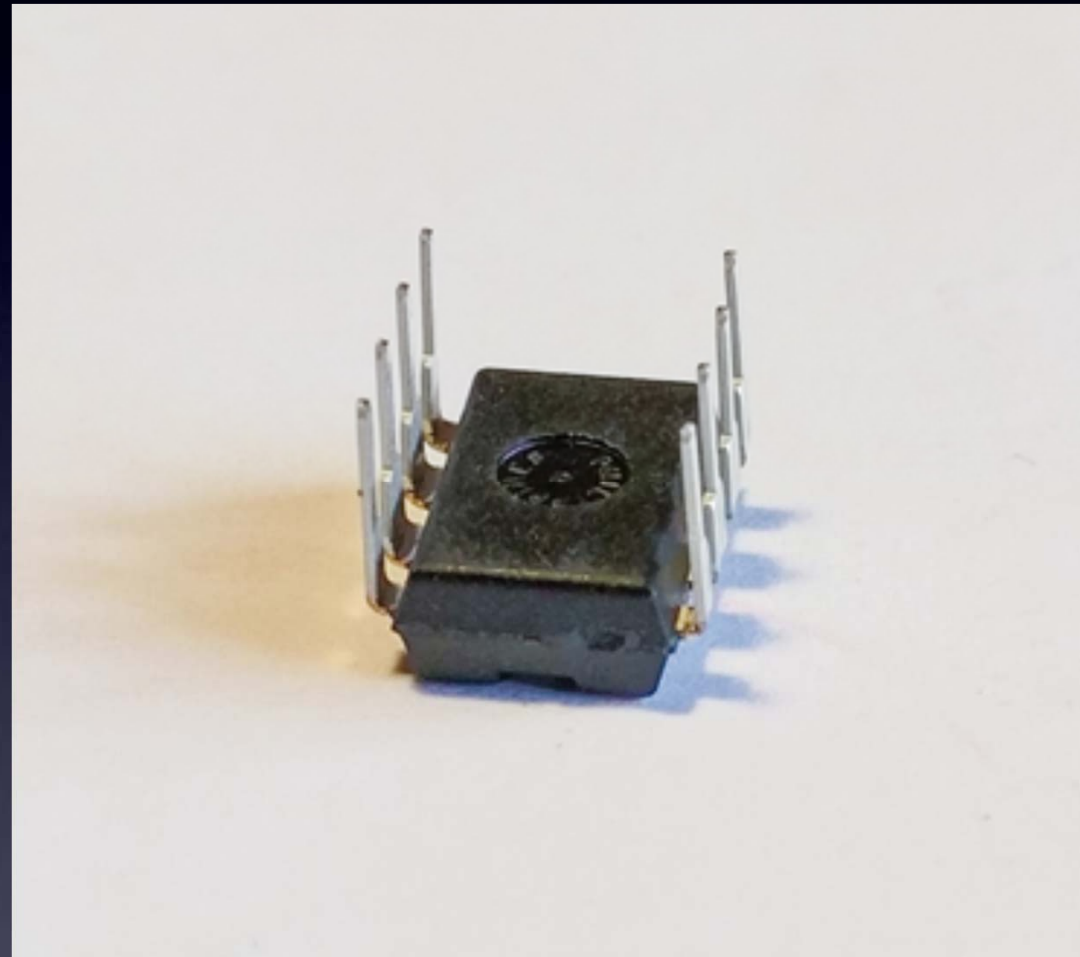
**U2**

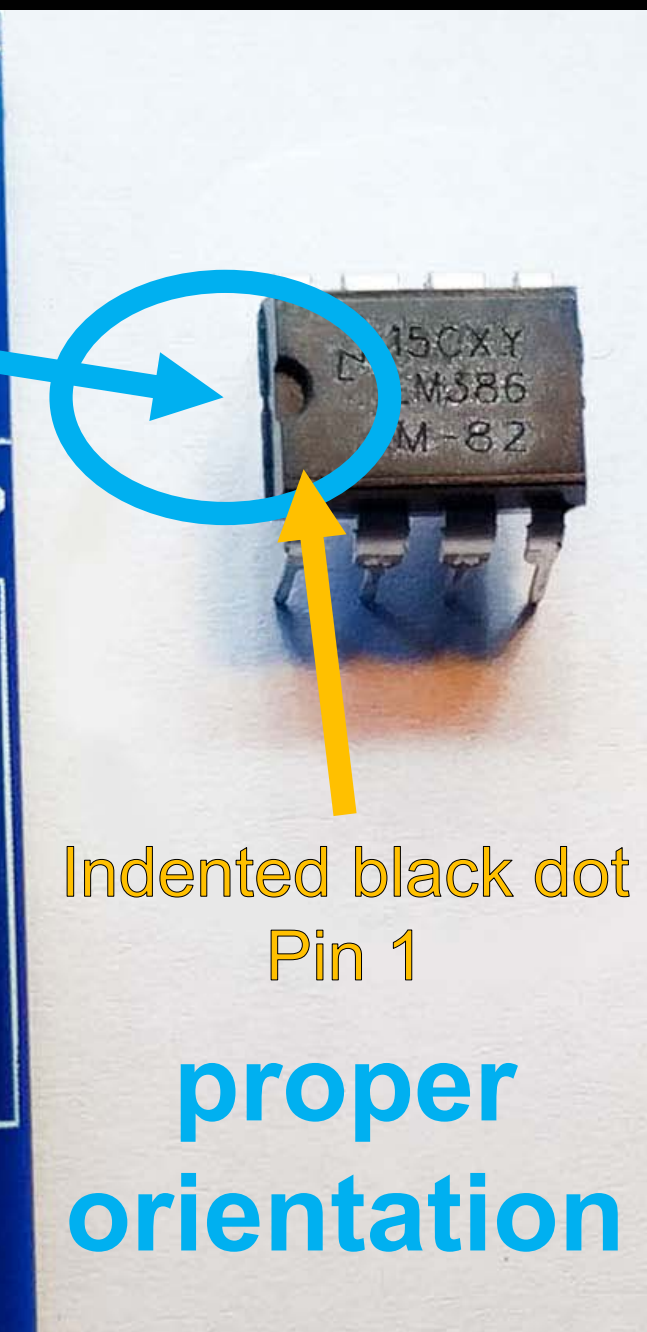
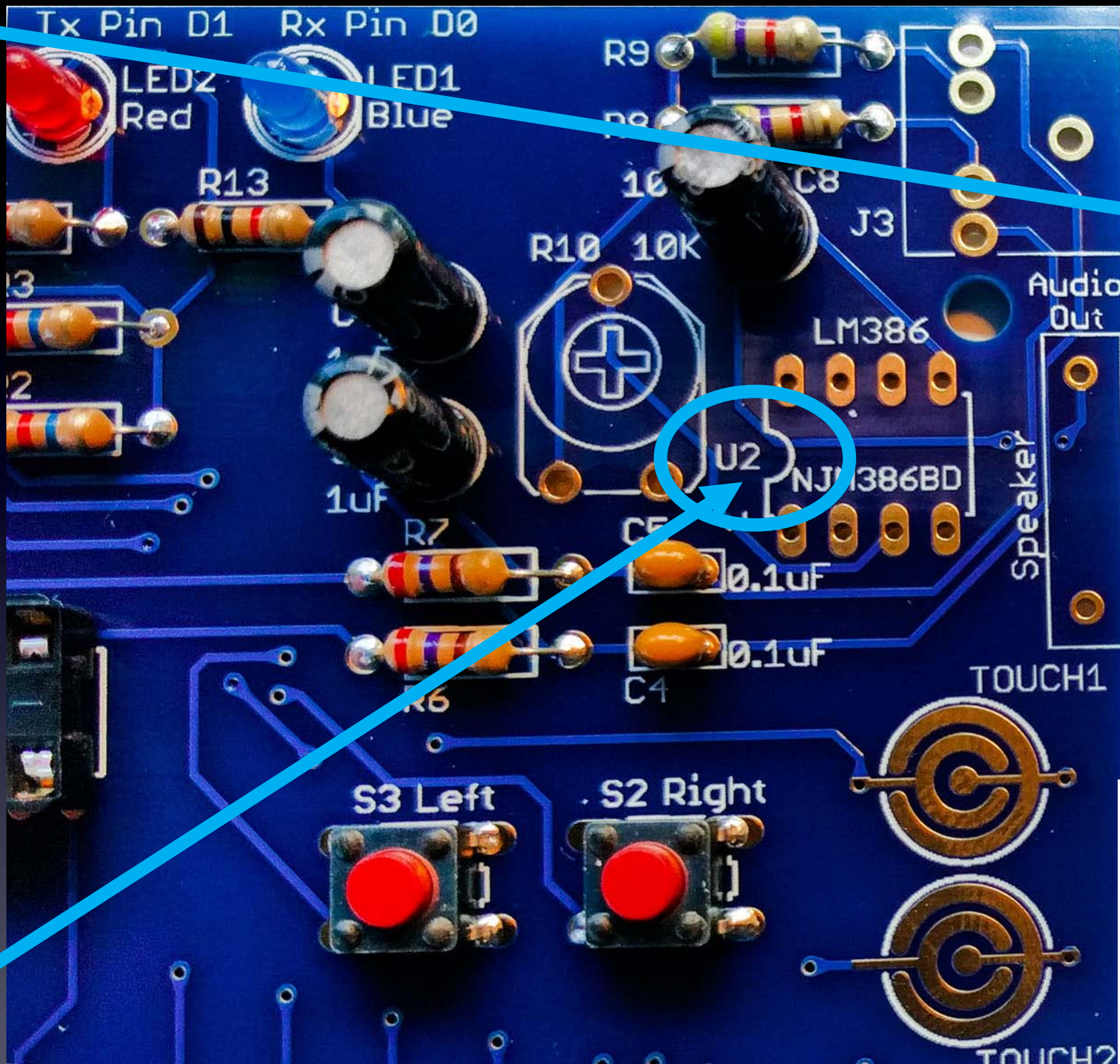


**We need the pins bent straight and parallel.  
Use your work table to (gently) bend the leads.**

# U2

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

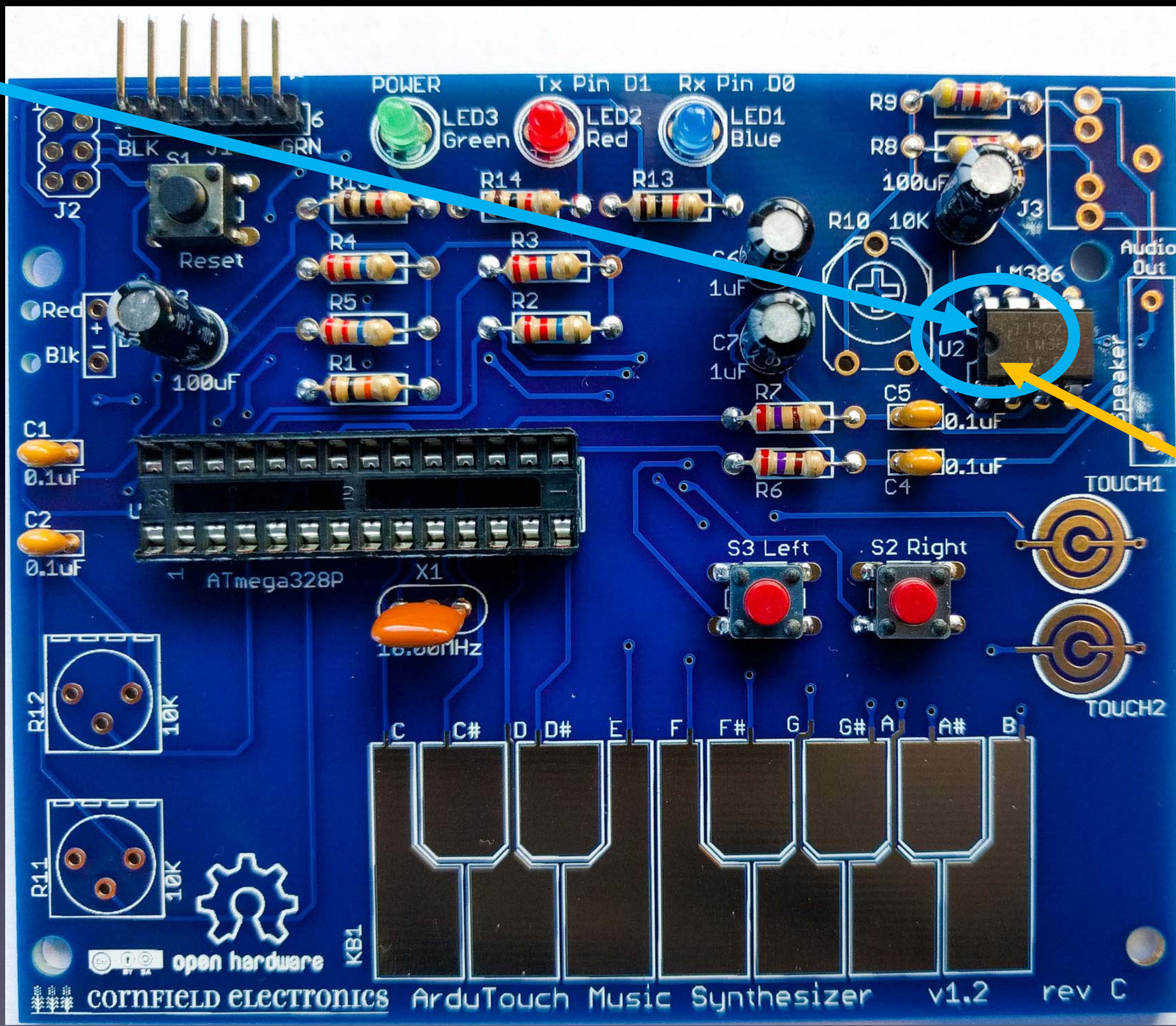




Indented black dot  
Pin 1  
**proper  
orientation**

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.

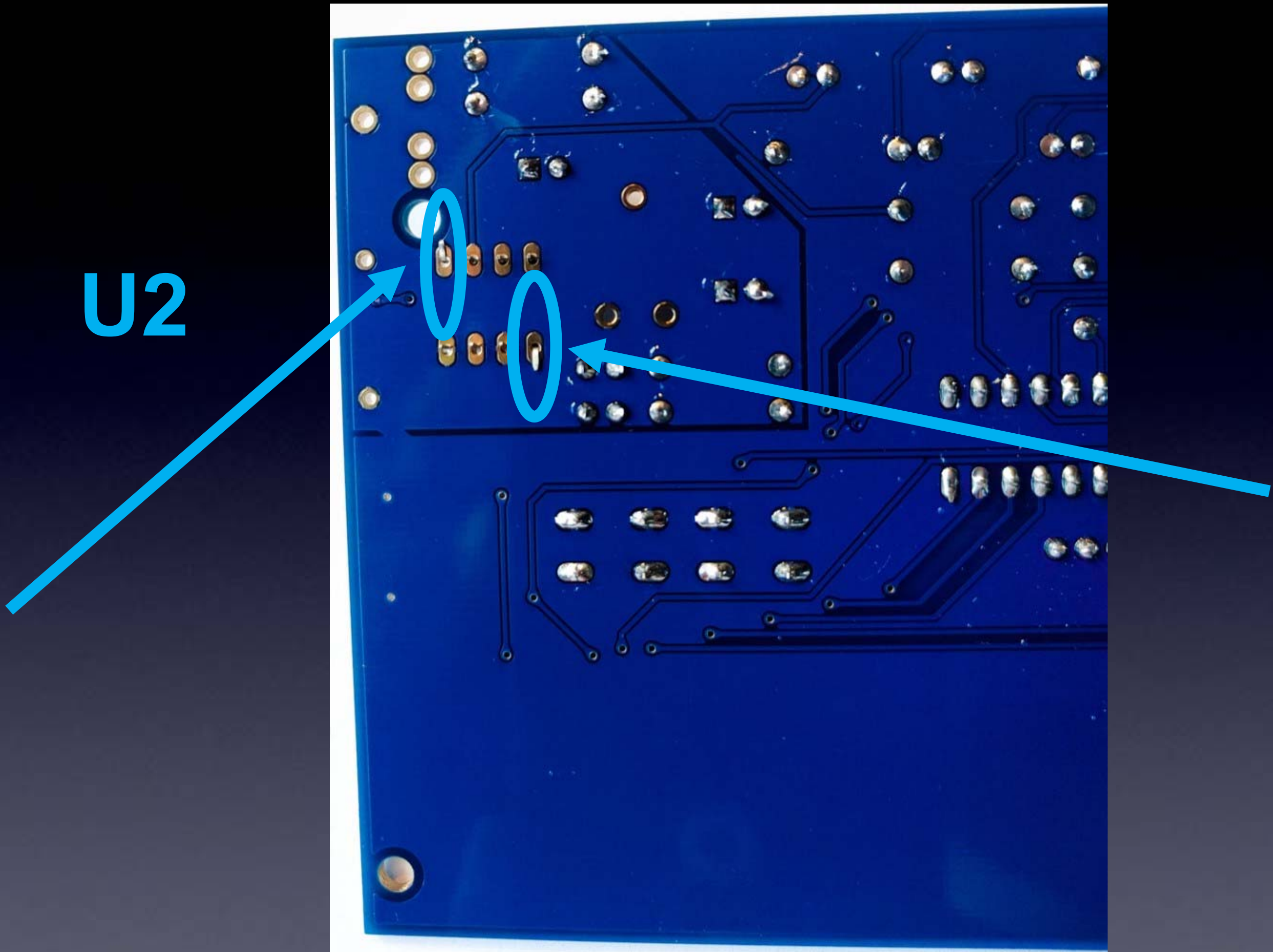
**U2: audio amp chip**



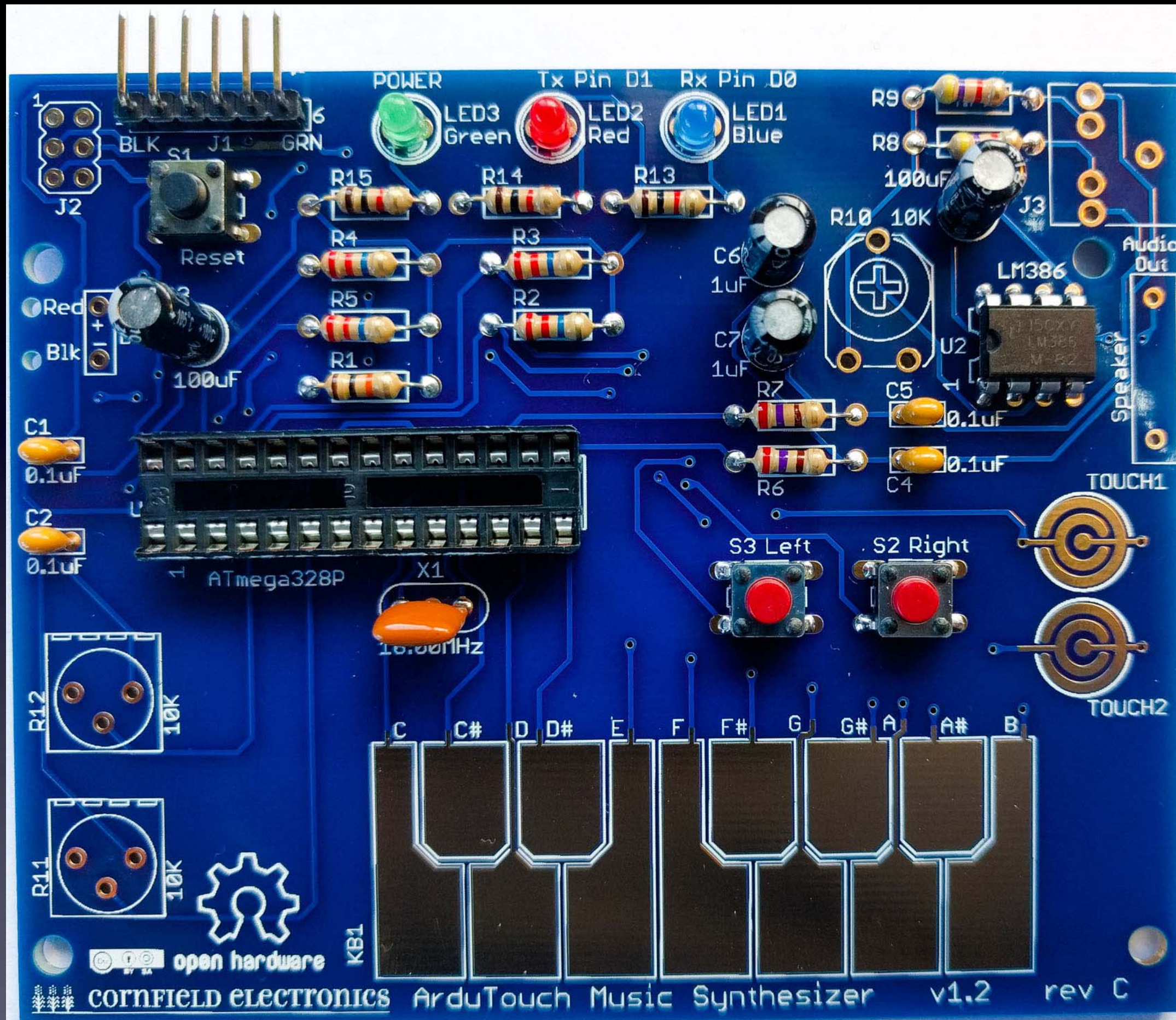
Indented black dot  
Pin 1

**U2: inserted correctly**



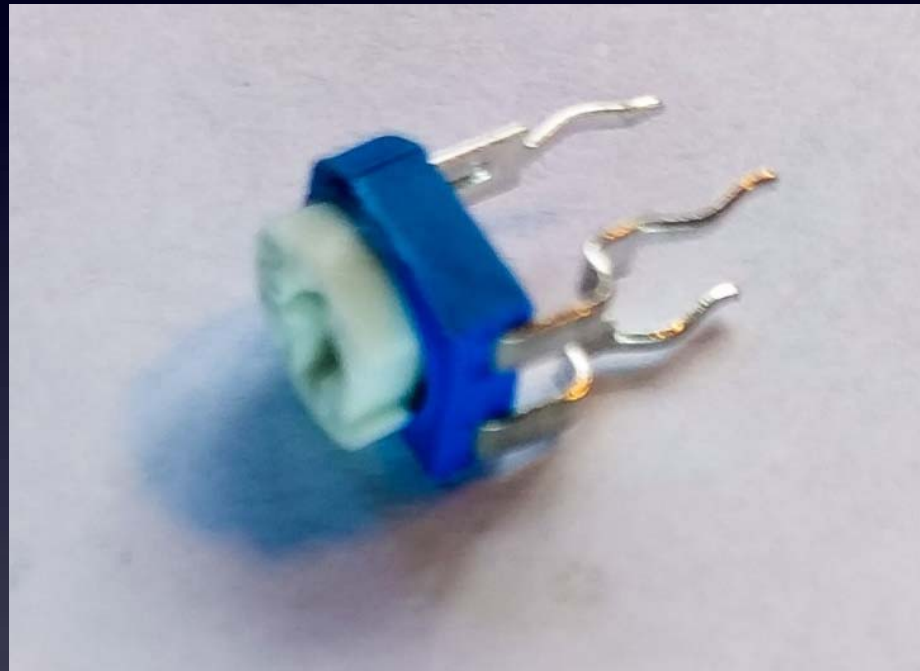


bend pins down on two corners,  
and solder all 8 leads to the board



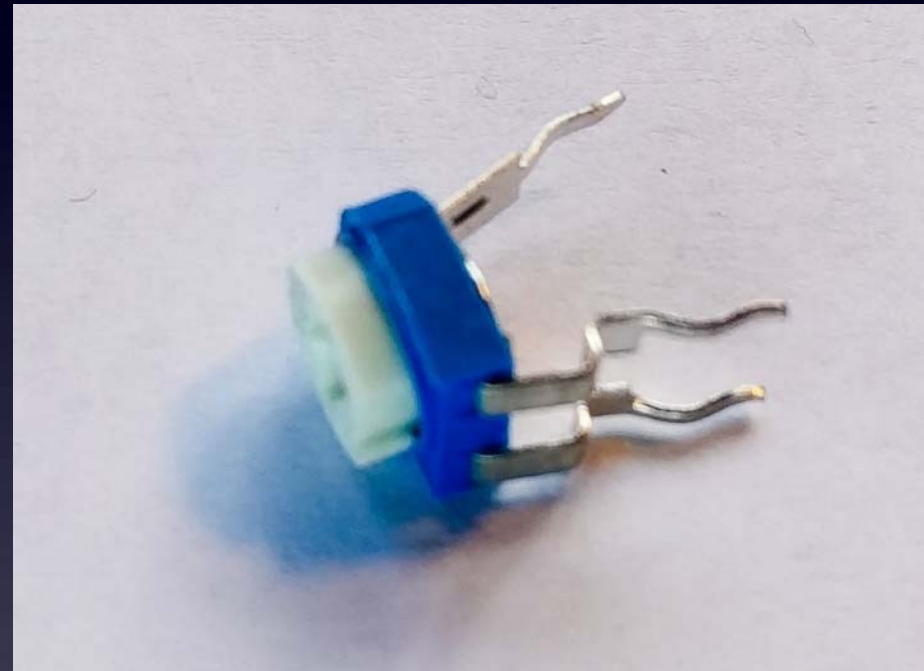
U2 – soldered to board

# R10: volume control



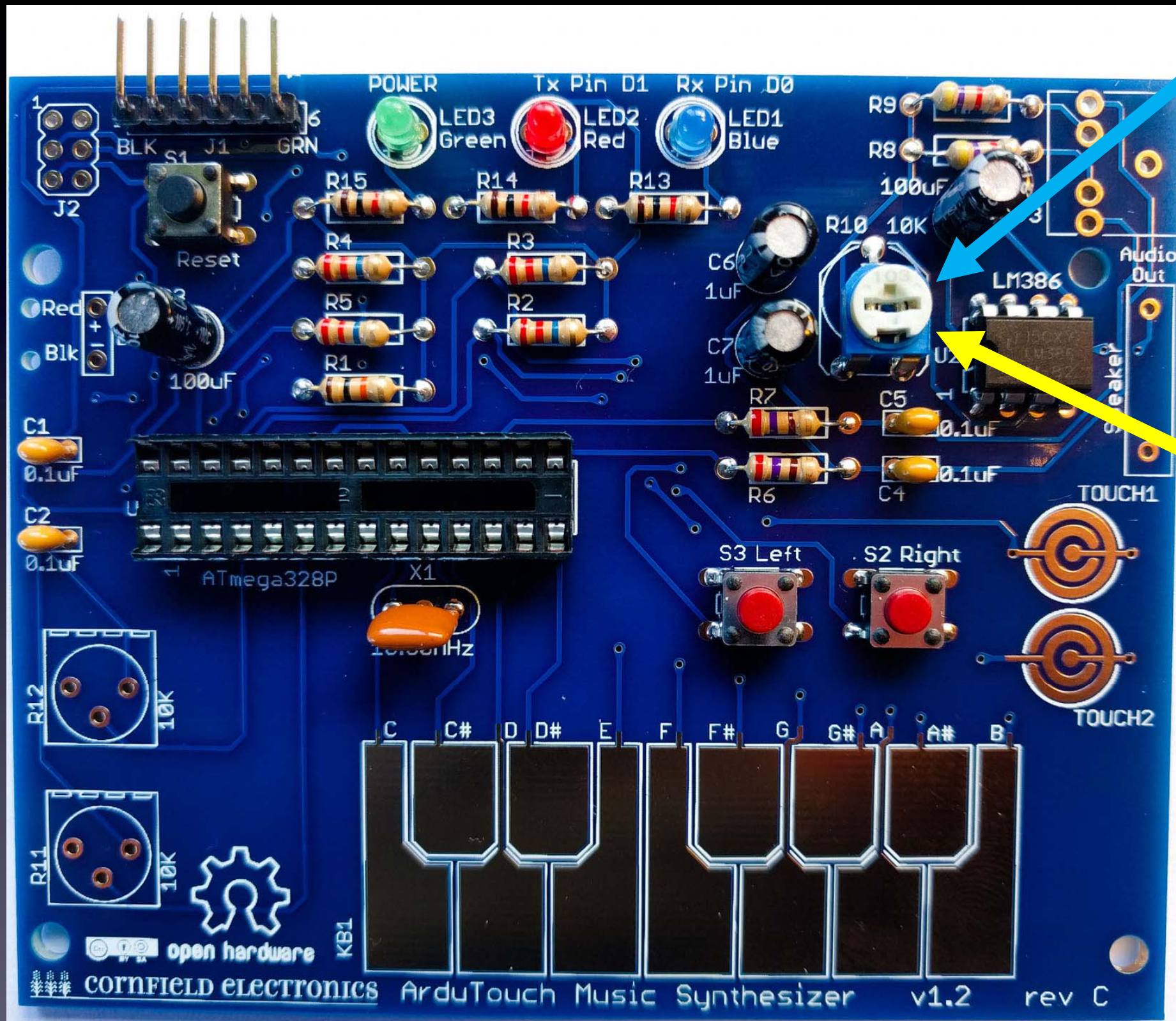
When new, the pins point straight down.

# R10: volume control

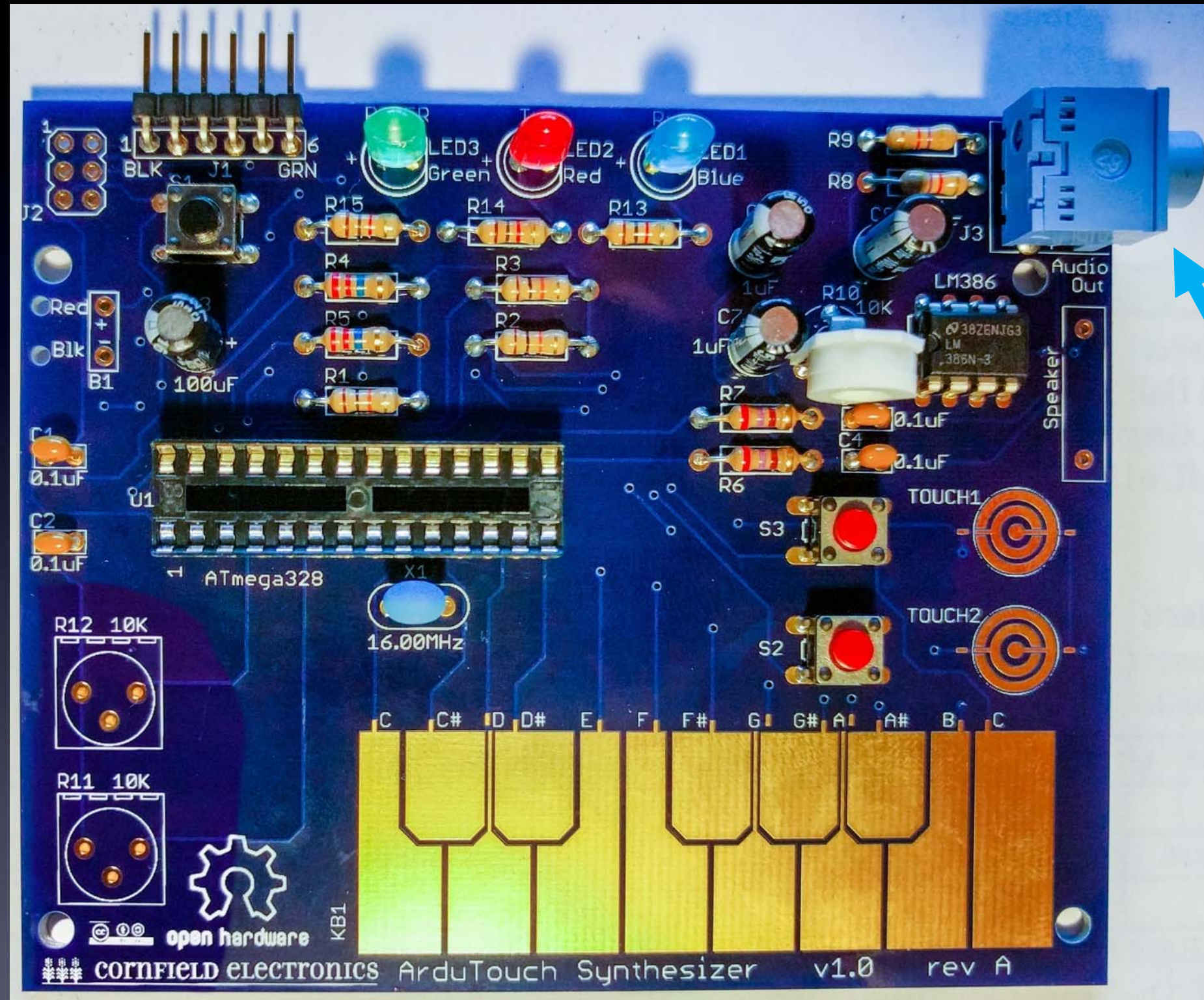


We need to bend them out a little to fit into the board.

# R10: volume control



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

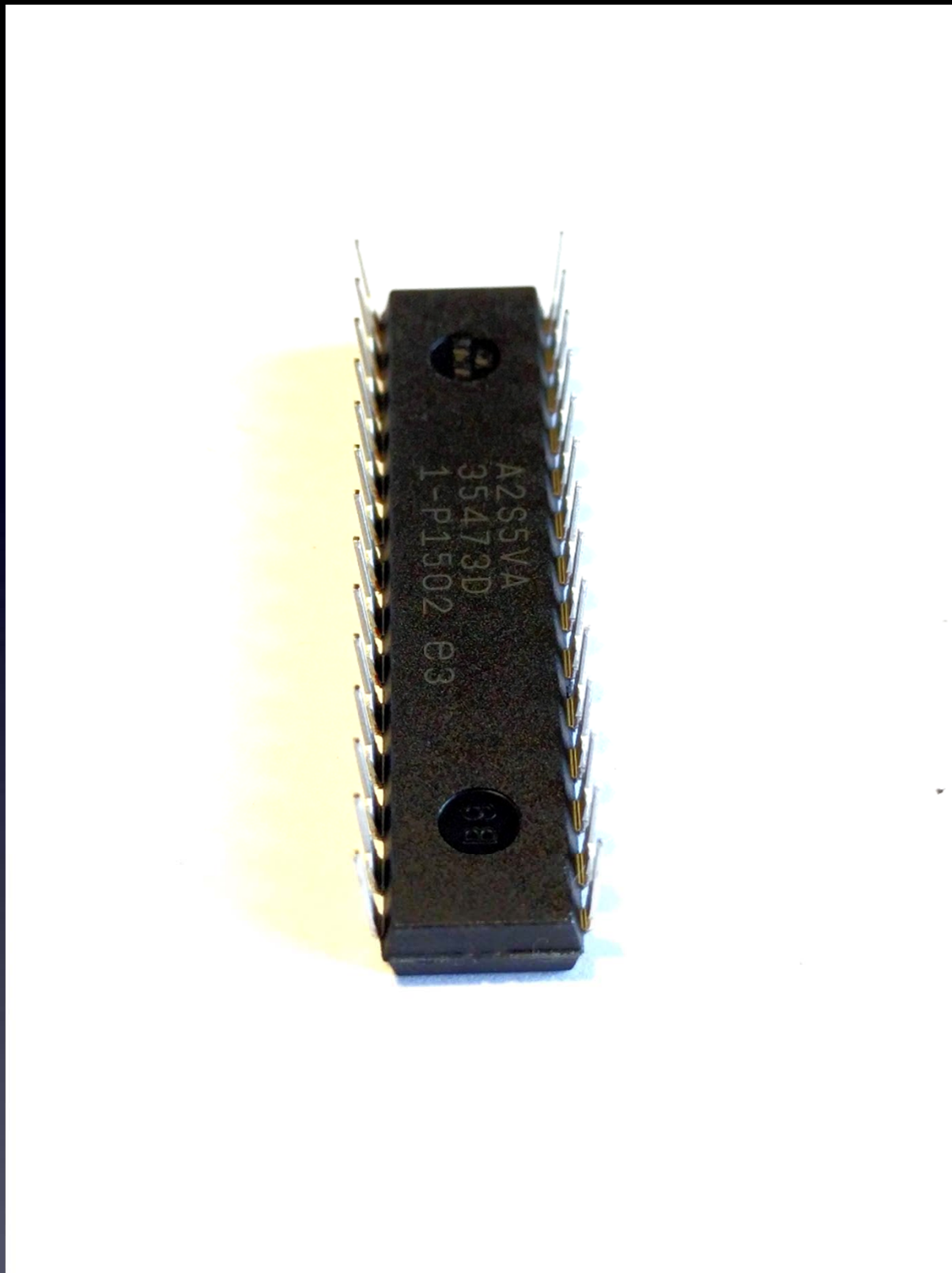


**J3: headphone / output jack**



**U1: microcontroller**

# U1



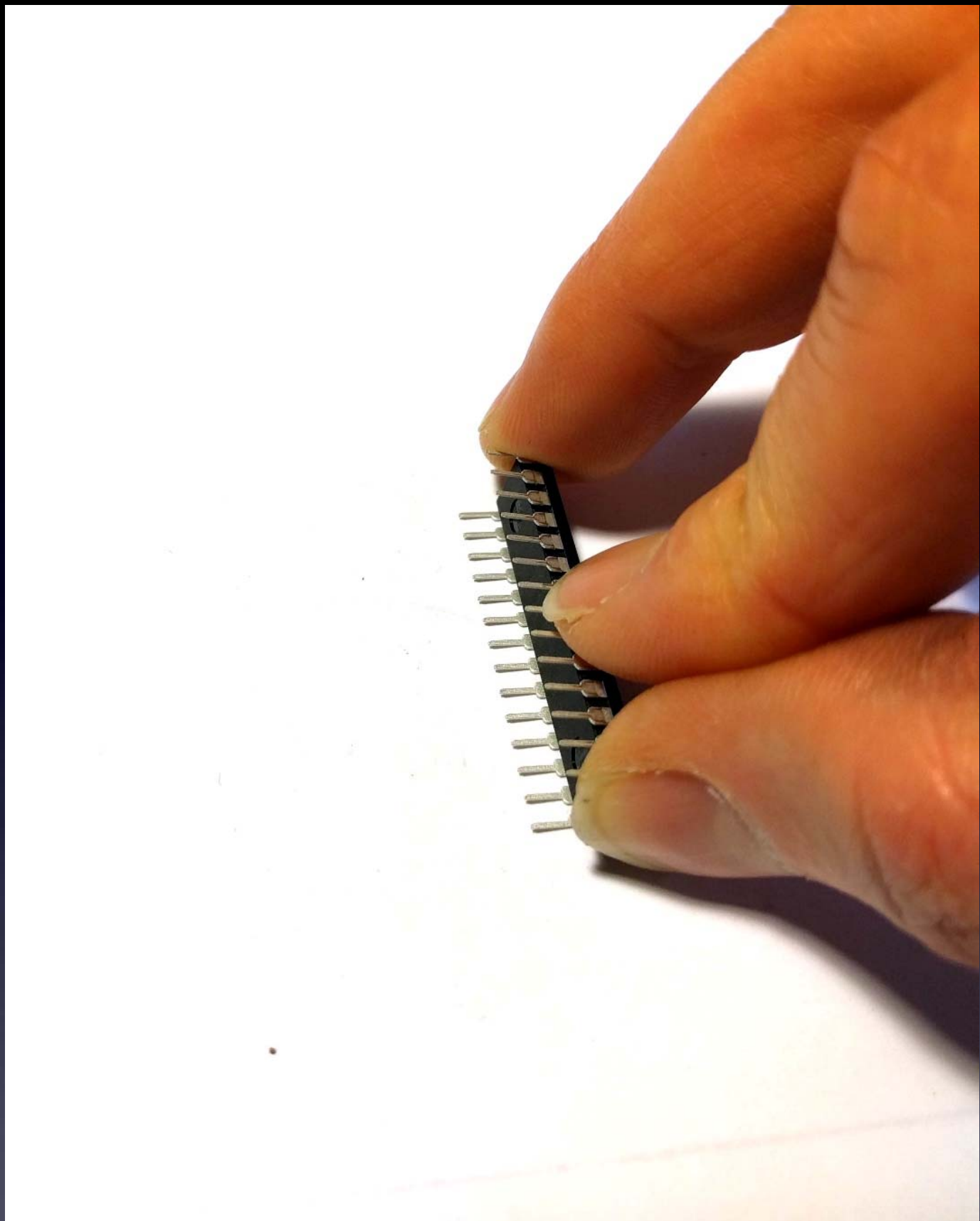
**When chips are new,  
their pins are bent out.**

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.

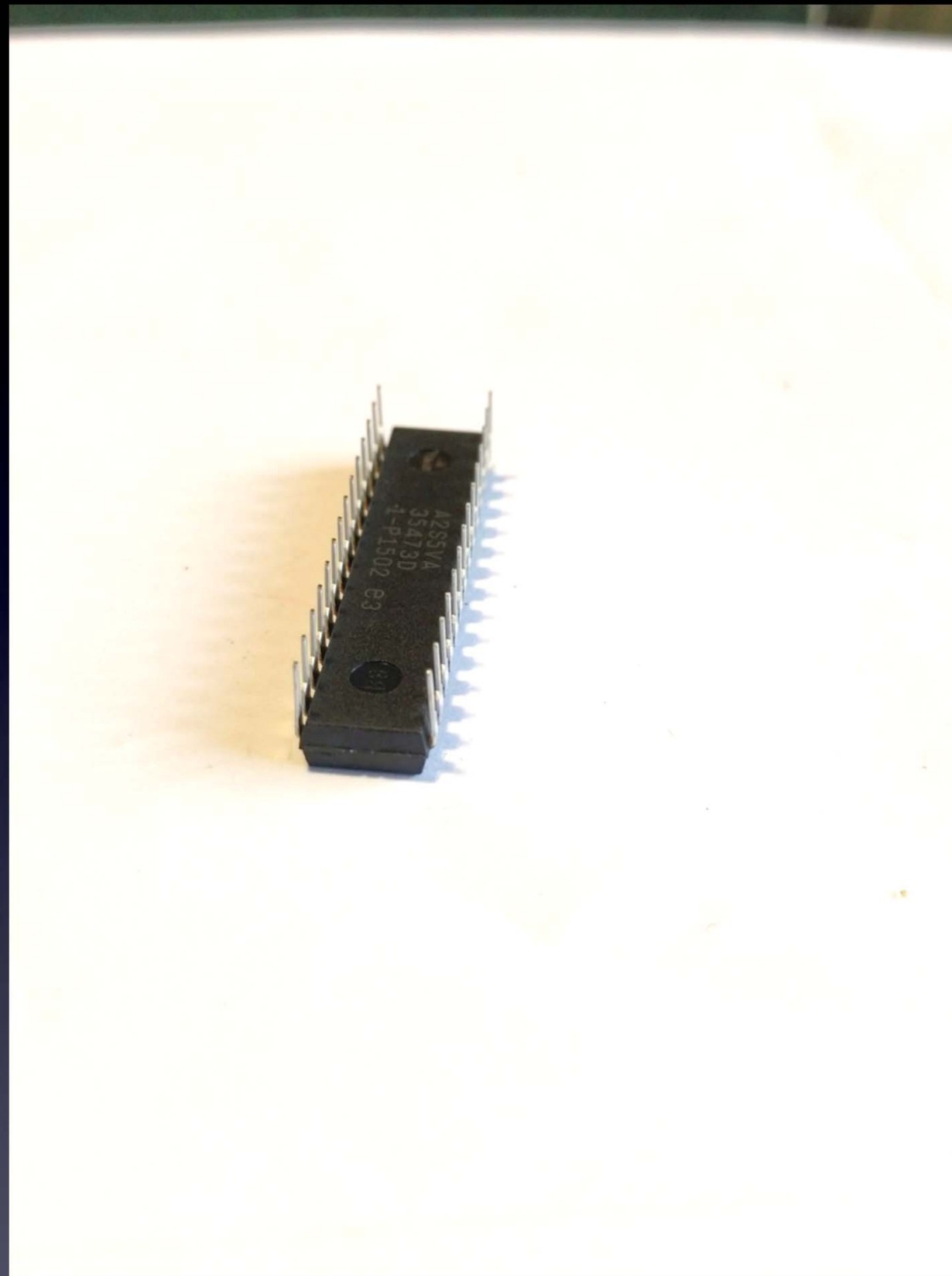


# U1

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.**

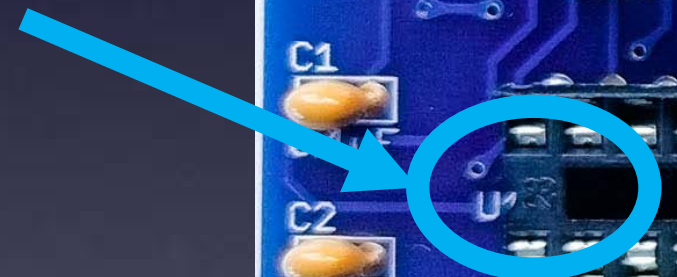
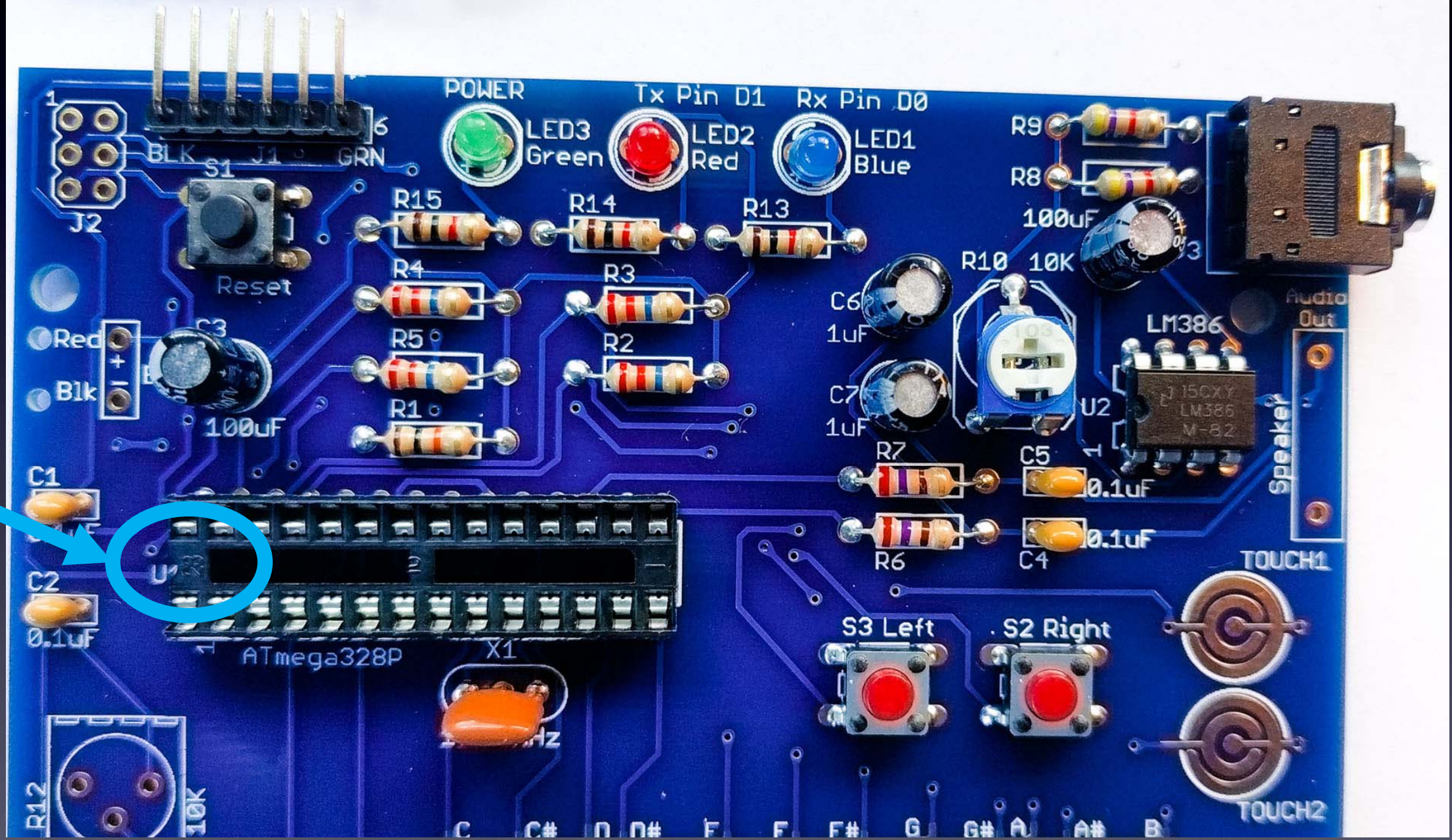


## **U1: microcontroller**

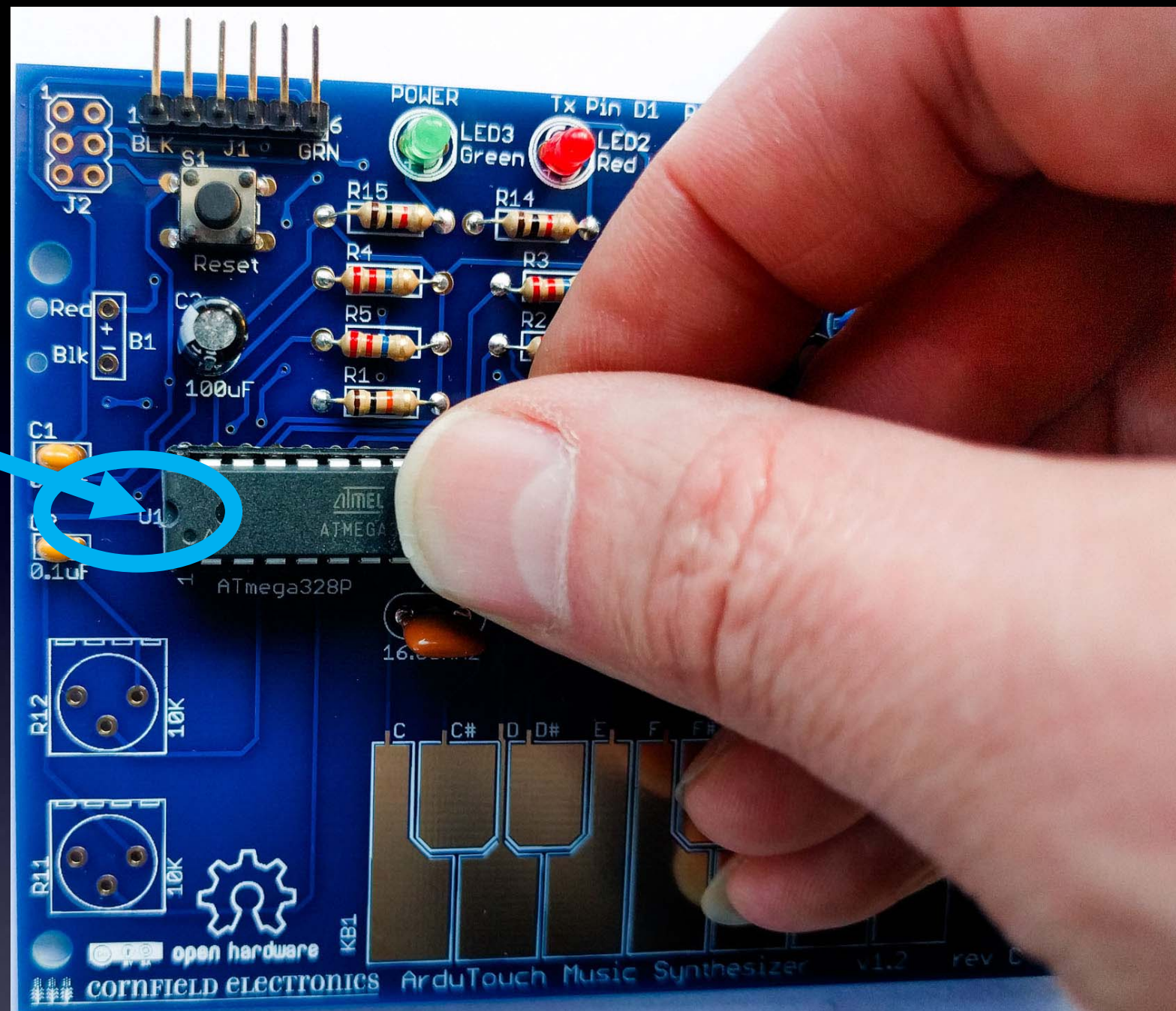
These pins must be straight and parallel



proper orientation



U1: microcontroller



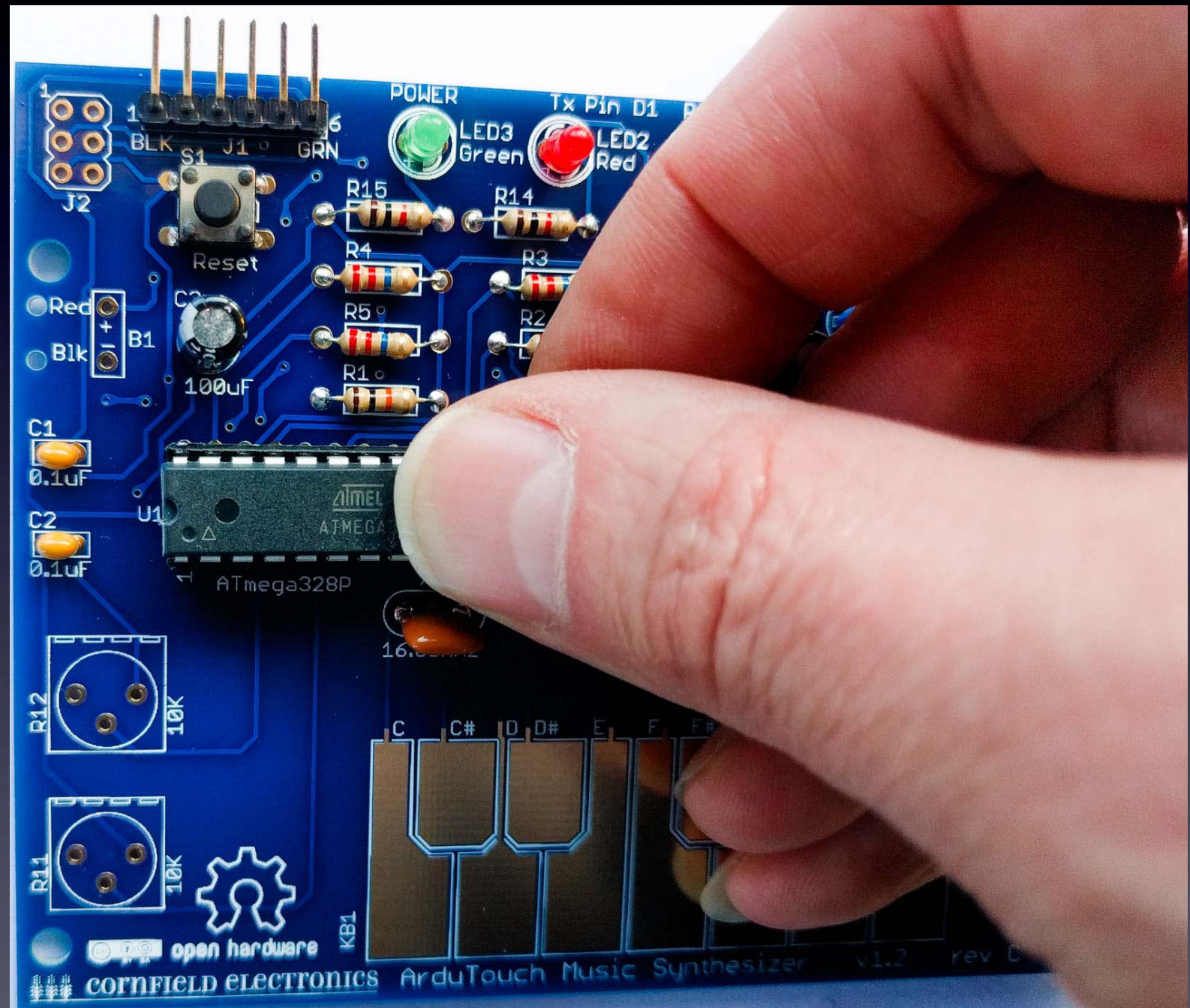
## U1: microcontroller

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

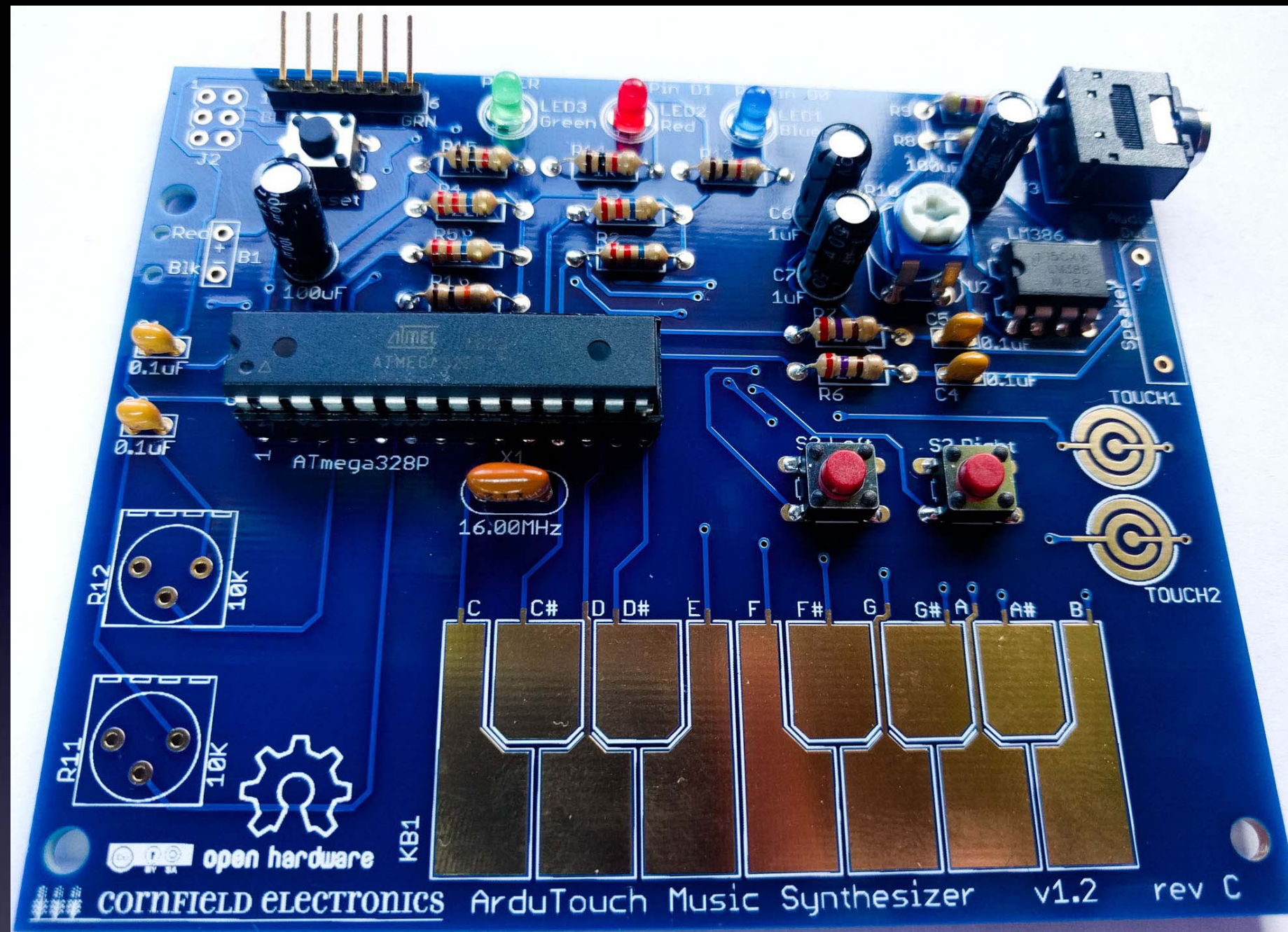
**Use two thumbs to push microcontroller into the socket**

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

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



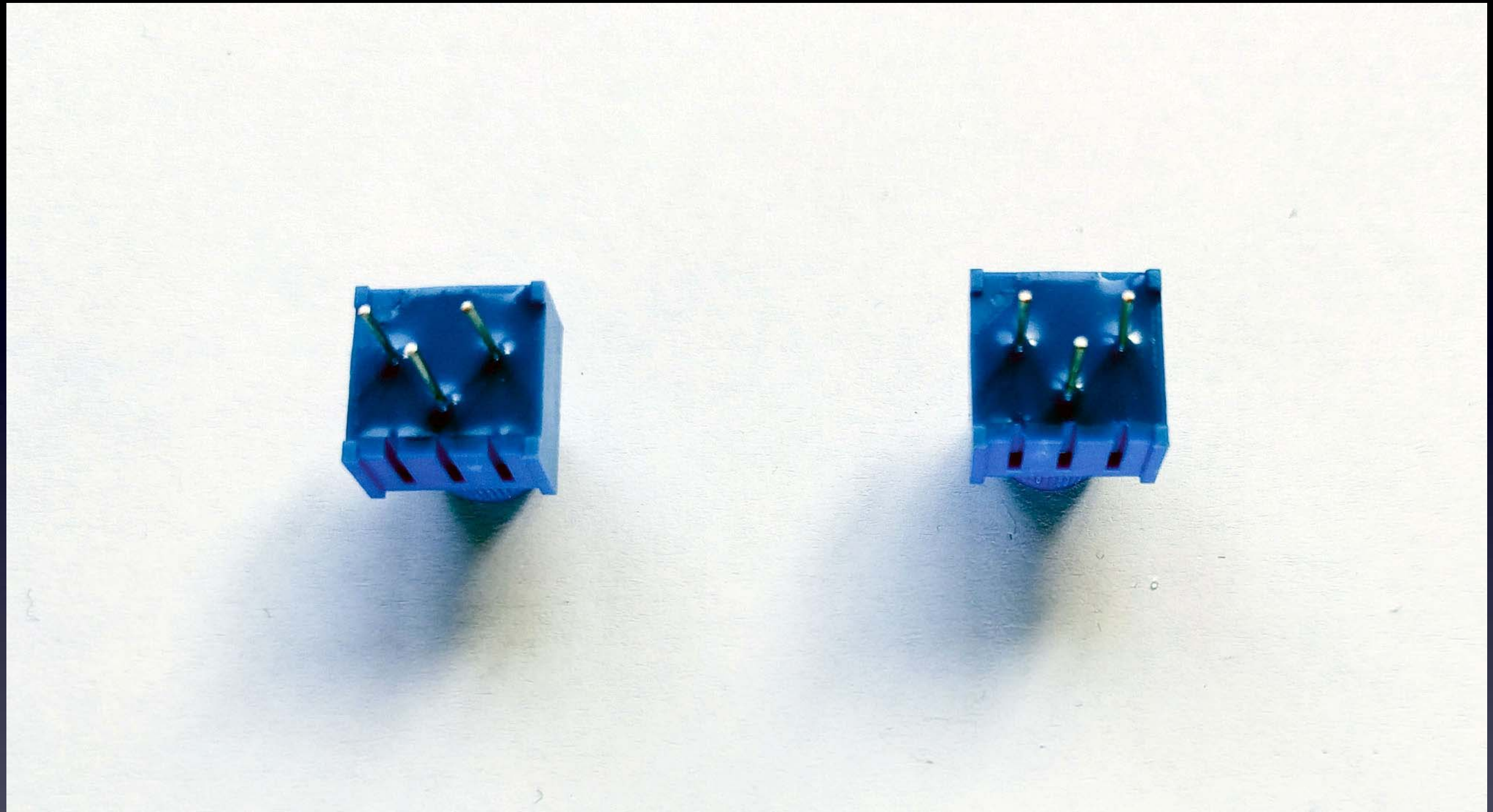
**U1: microcontroller**



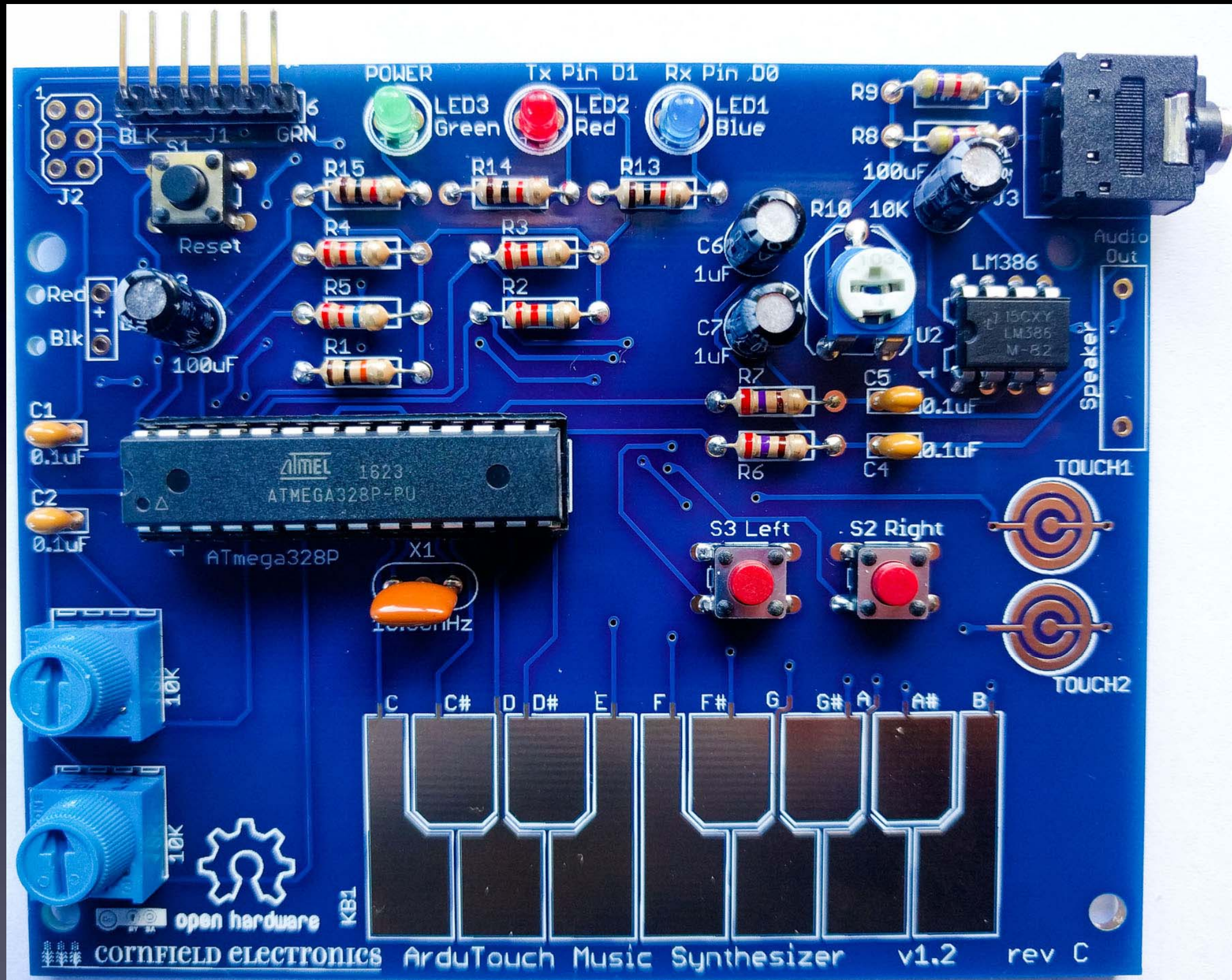
## U1: microcontroller

**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**





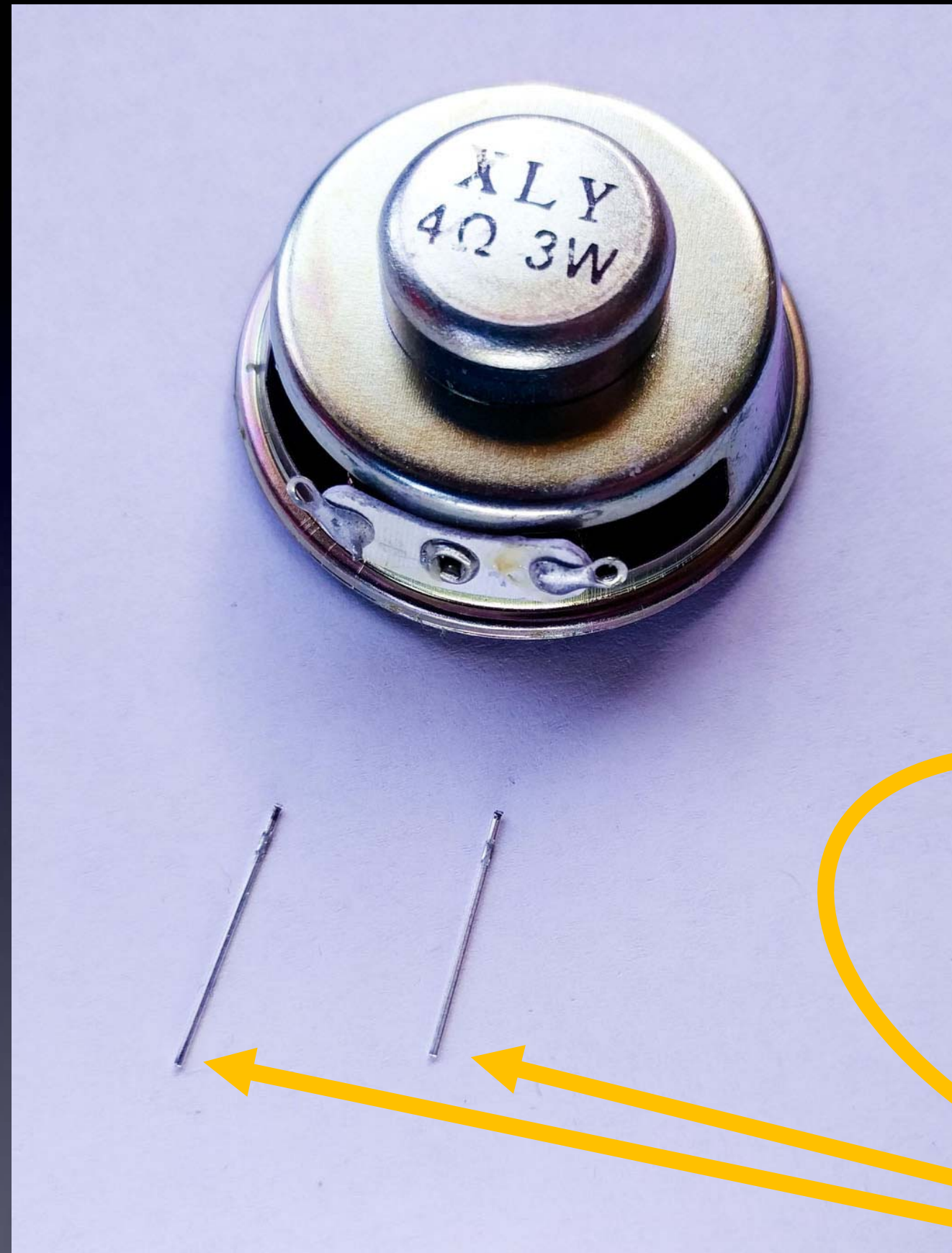
**Speaker**

**Some kits have a speaker that looks like this**



**Speaker**

We'll add leads  
to the speaker



Saved  
leads

from the LEDs

Speaker

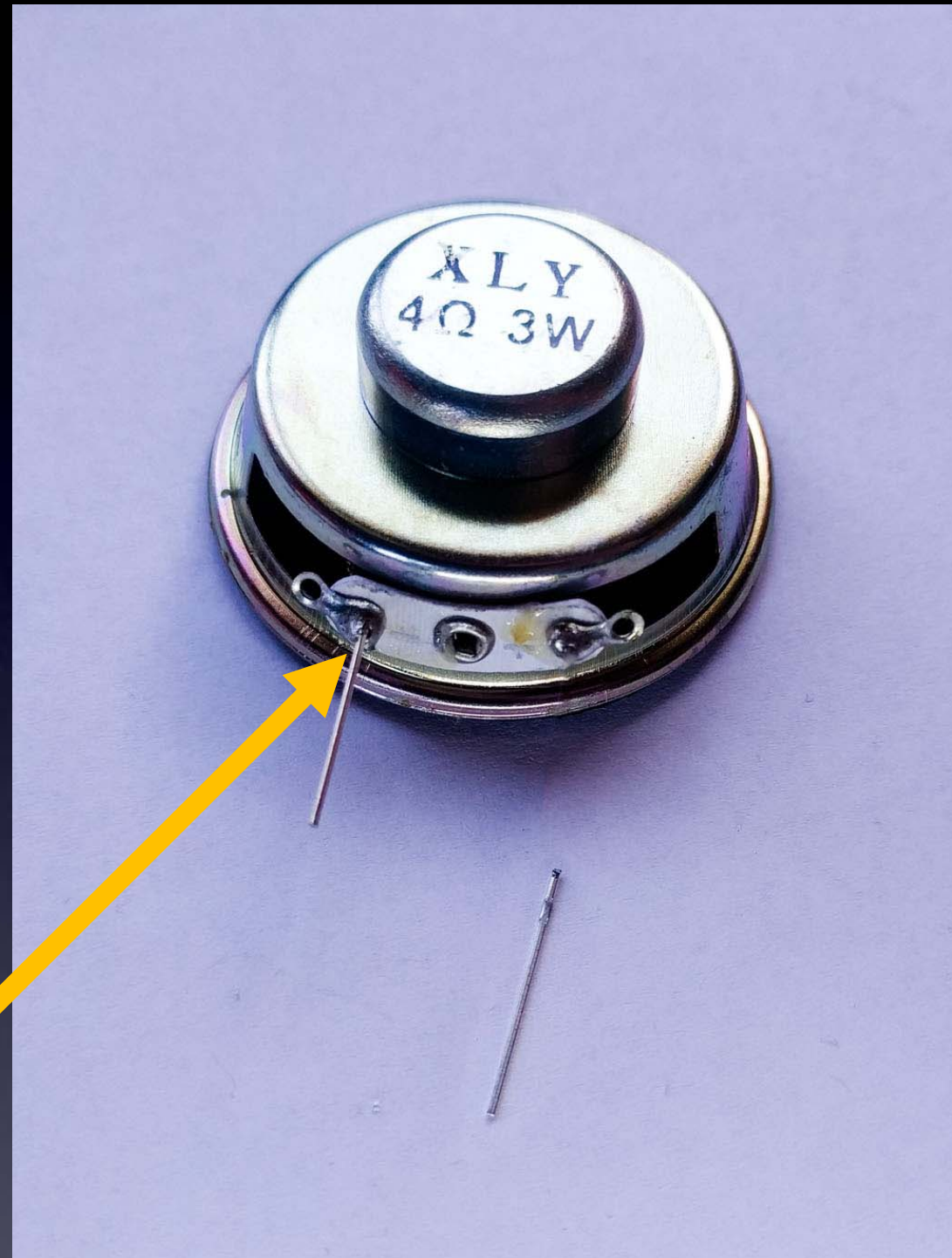
**Tin one side  
of each lead**

(i.e., cover with  
thin film of melted solder)



**Speaker**

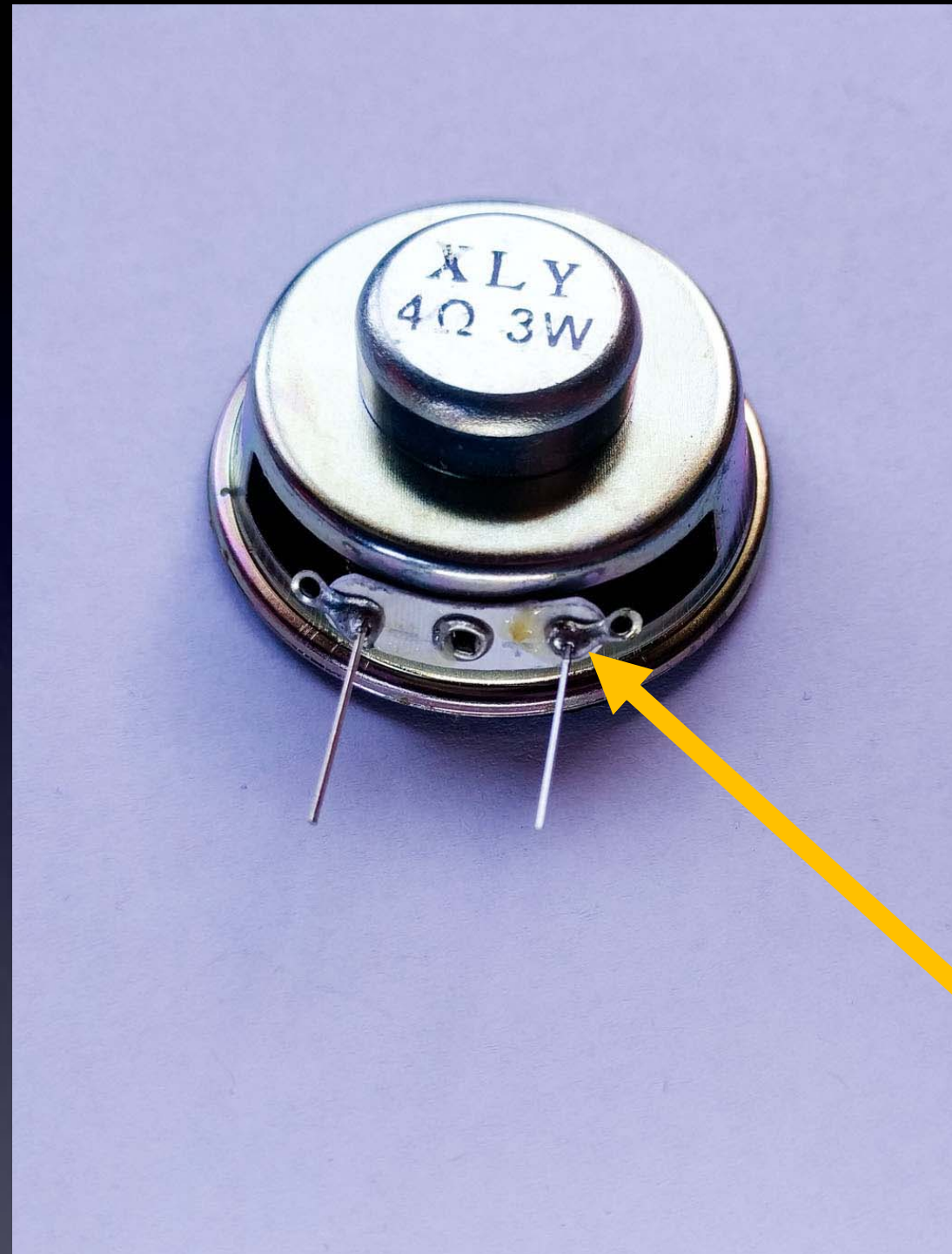
**Solder one lead  
to speaker**



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

**Speaker**

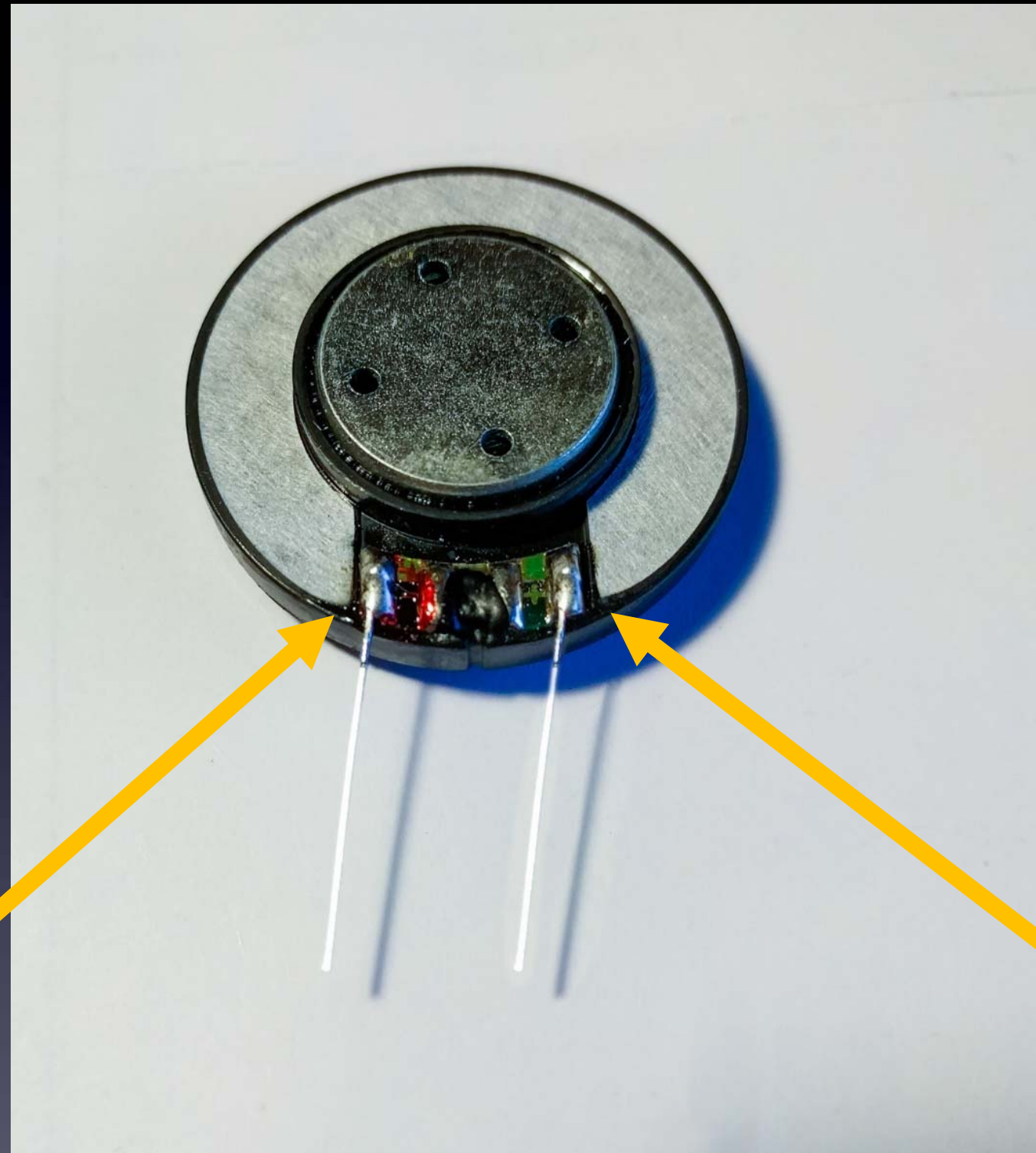
**Solder next lead  
to speaker**



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

**Speaker**

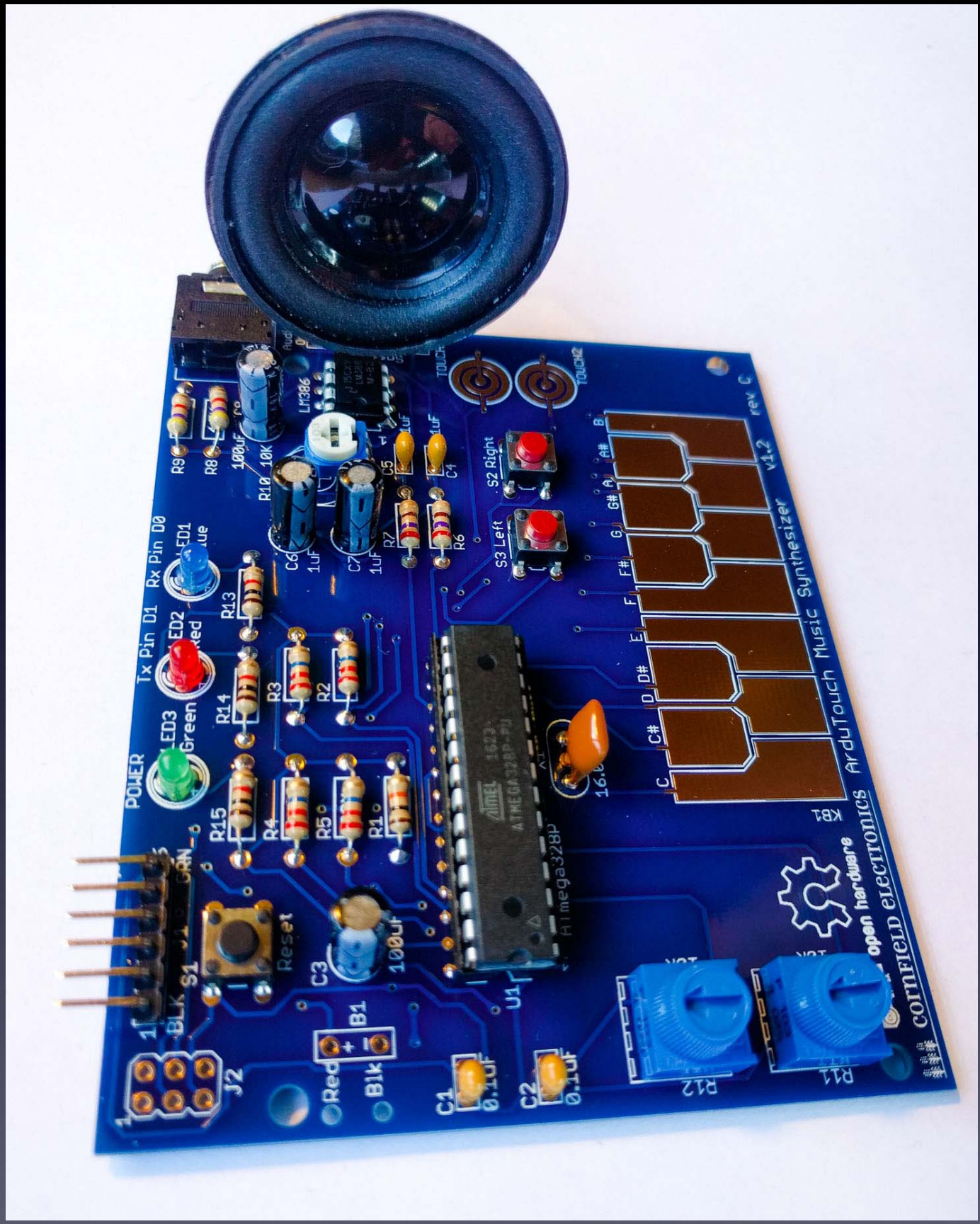
**Some kits have a speaker that looks like this**



**Notice the correct place to solder the wires**

**Speaker**

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

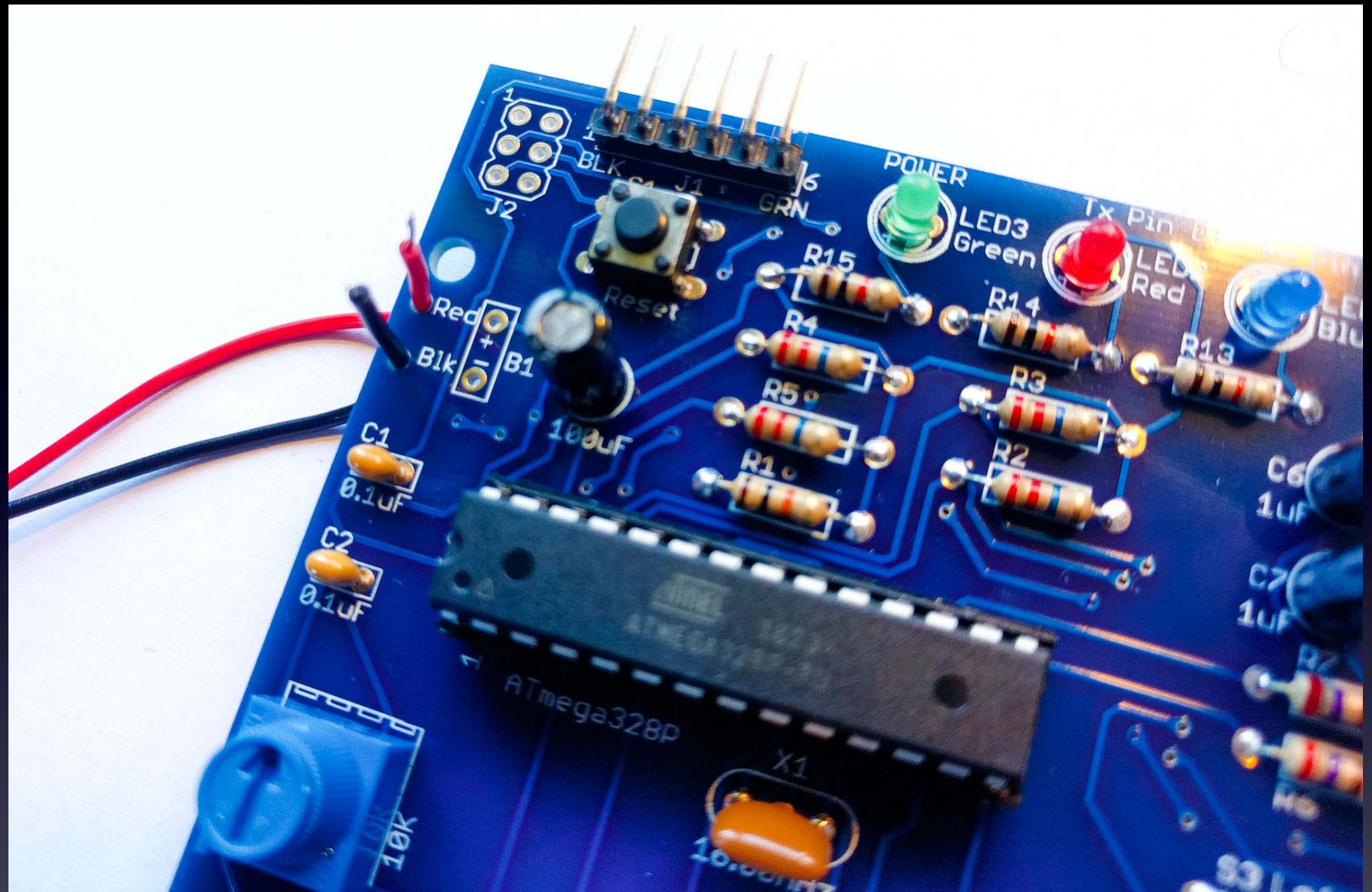


**Speaker**



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

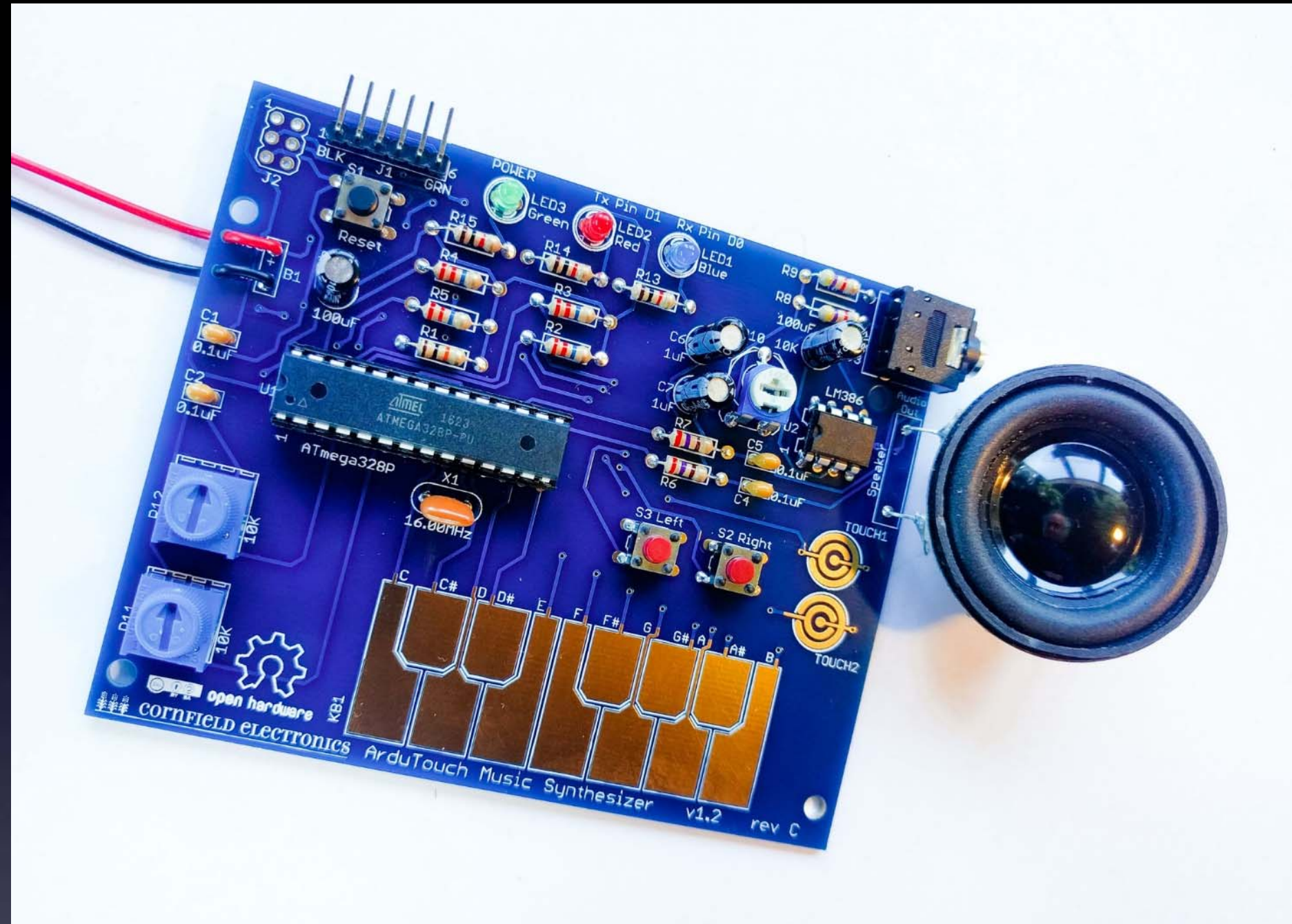
*If so, you can widen these two holes by gently rotating a scissors or small knife or small Phillips screwdriver on the top and bottom of these two holes.*



**Push battery pack leads through holes.**

**Make sure Red and Black go through their correct holes!**

**Battery pack**

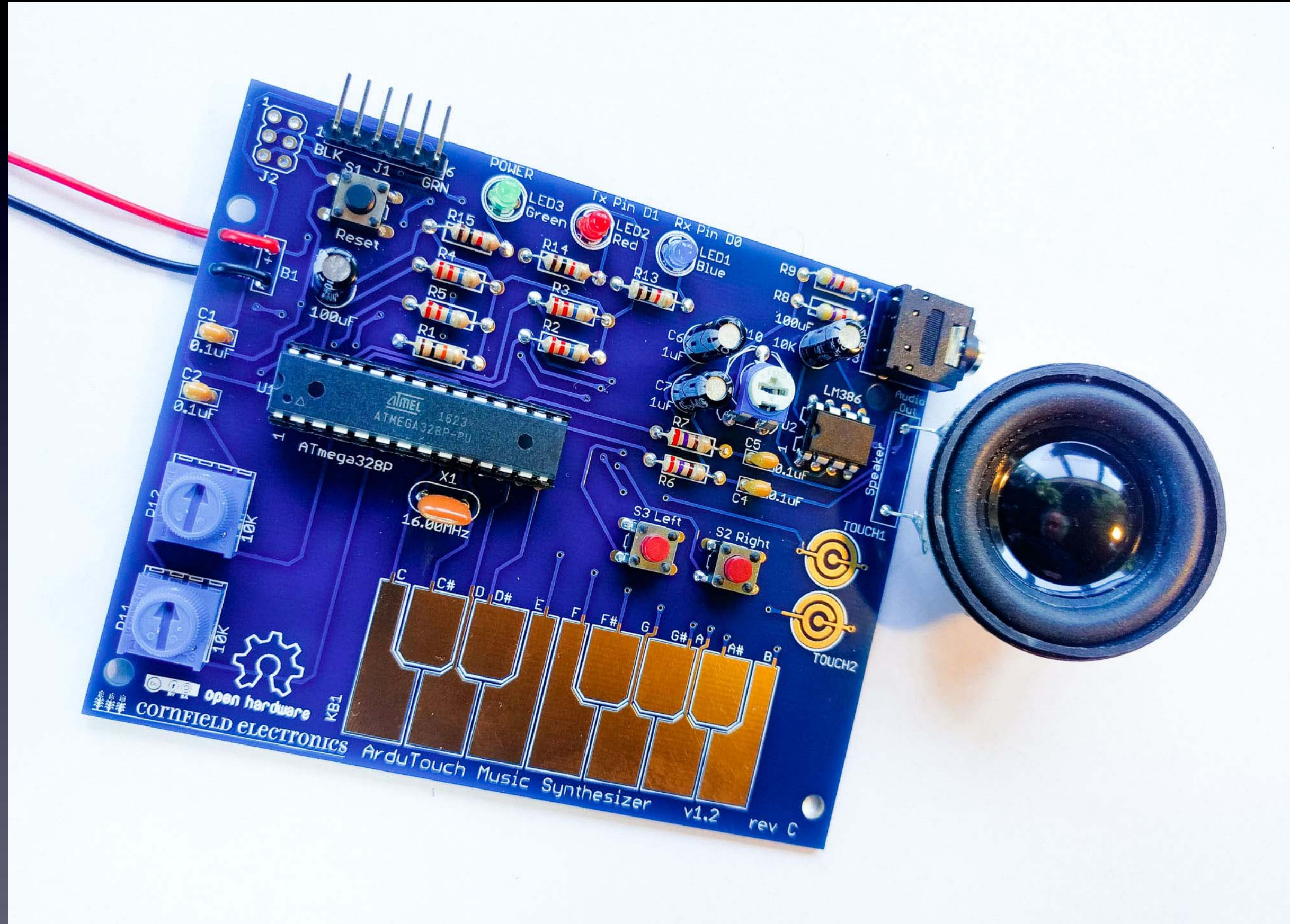


Loop one lead into its pad,  
and solder.

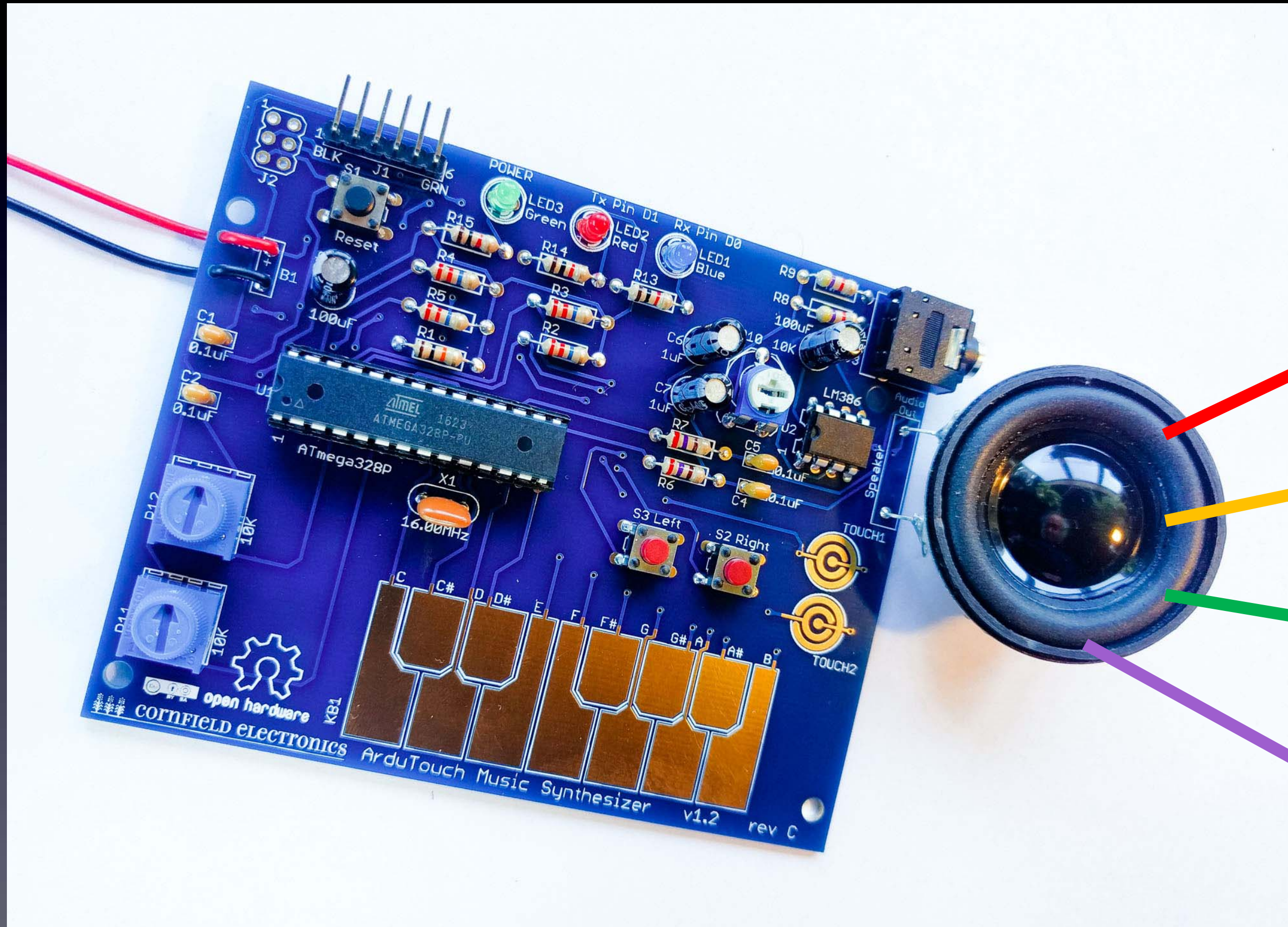
Then loop the other lead into its pad,  
and solder.

**Battery pack**

# Done!



# Let's make noise!

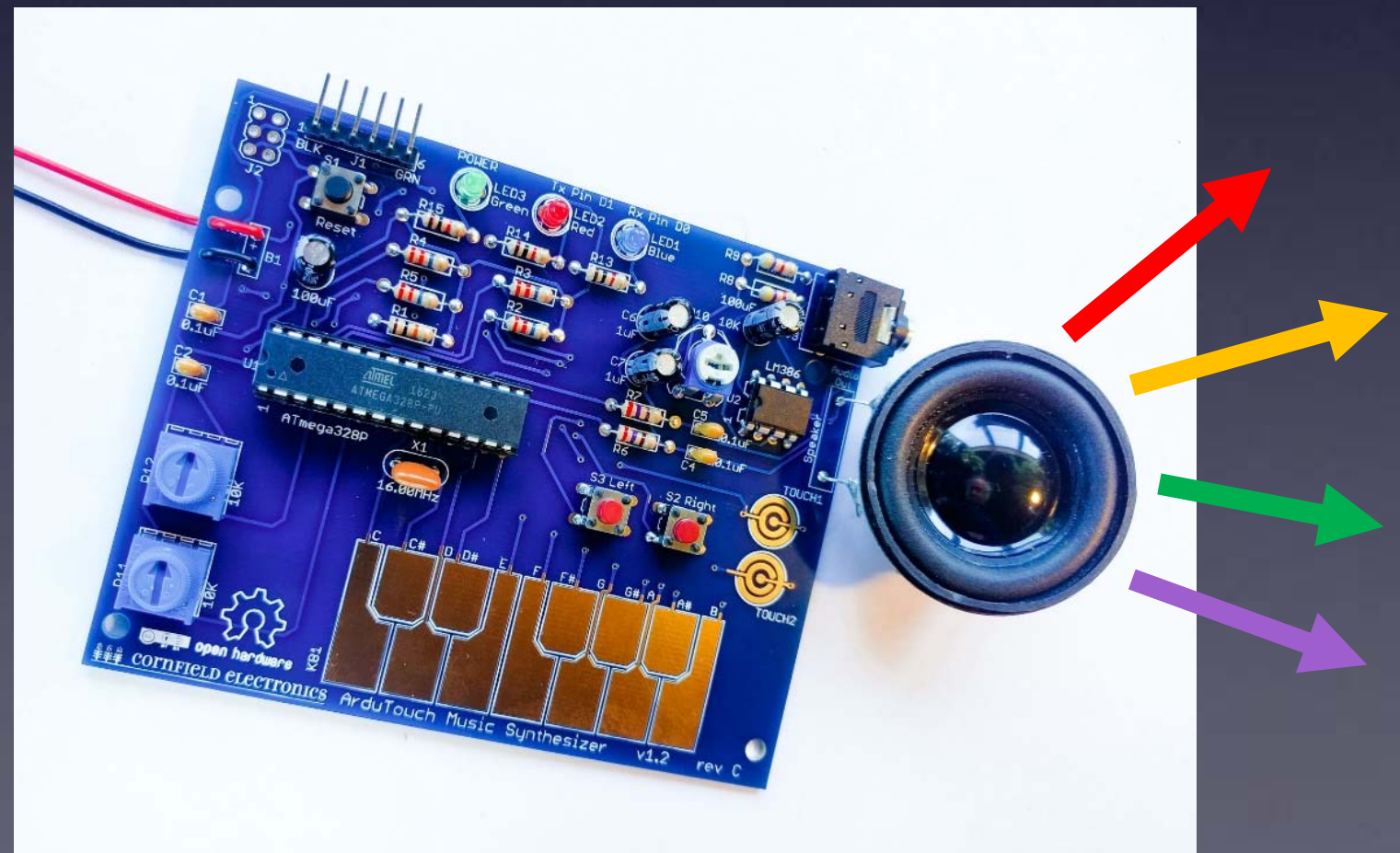


# Let's make noise!

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

"Thick" plays 4 sawtooth waves 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!

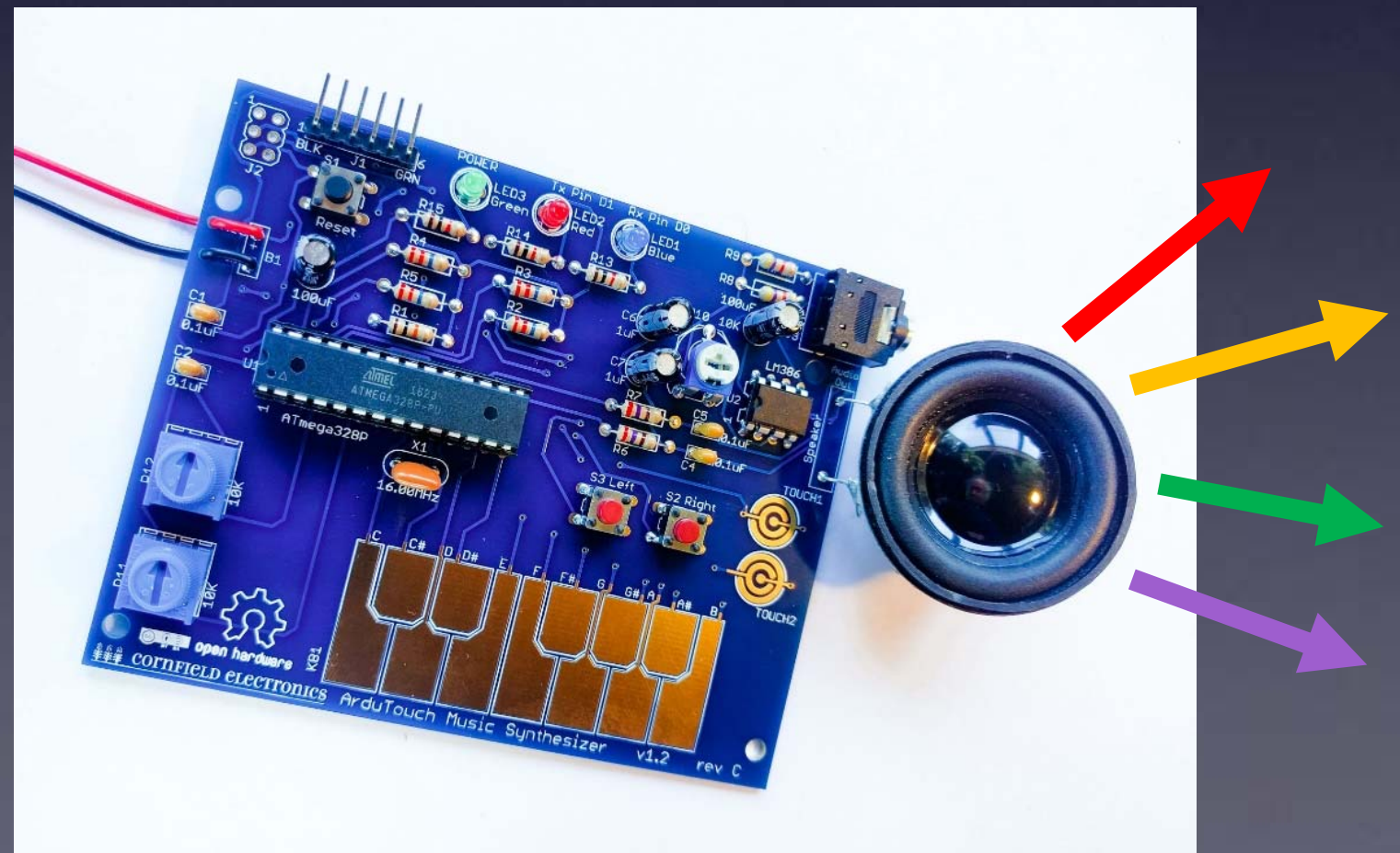


# Let's make noise!

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...

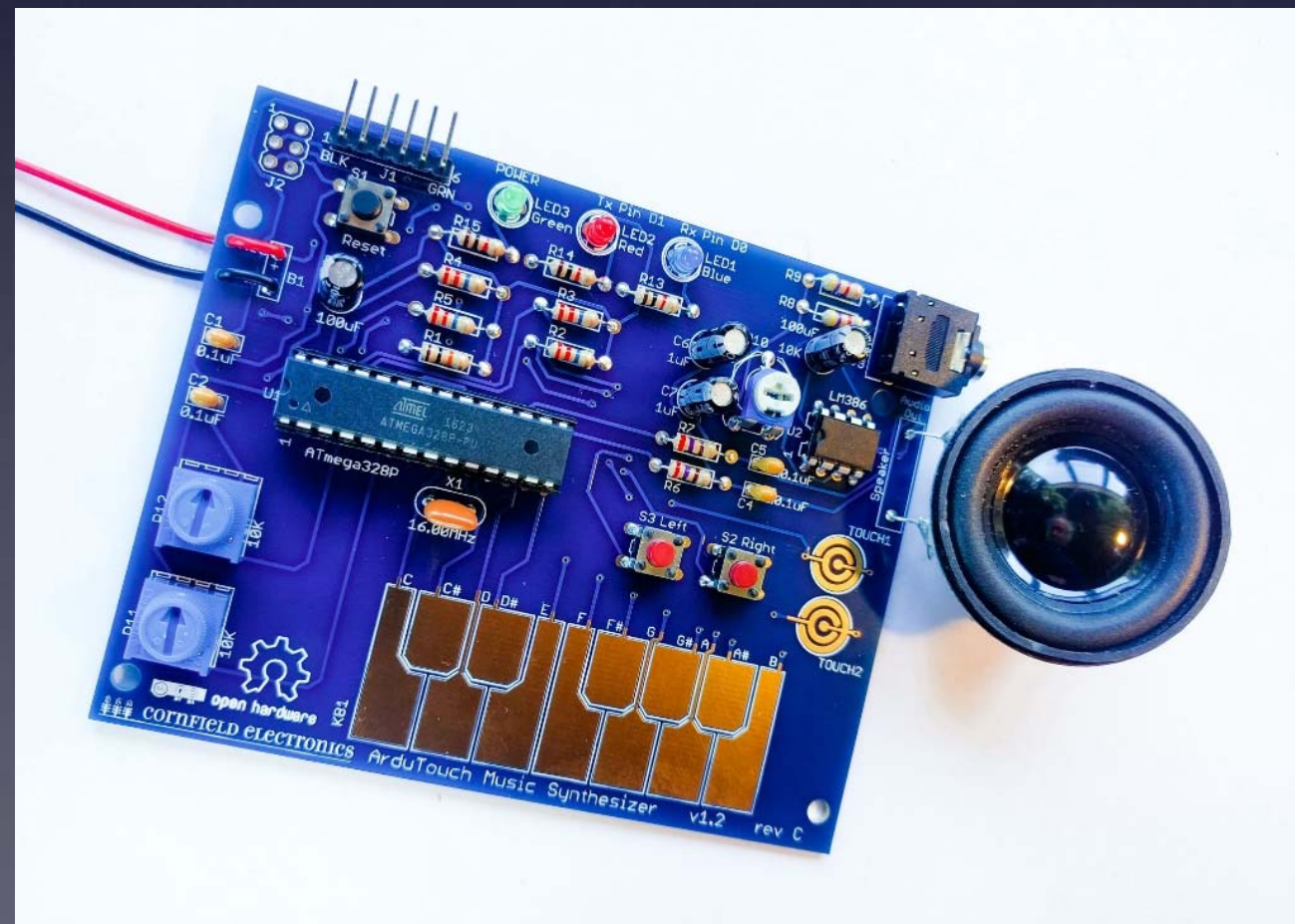


# Re-programming the ArduTouch

**We have written several way cool synthesizers for 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://cornfieldelectronics.com/cfe/products/buy.php?productId=usbcable>>
- a synth sketch and the ArduTouch Arduino library  
<<http://cornfieldelectronics.com/cfe/projects.php#ardutouch>>



# 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>>**





# Arduino

## First:

Download and install the Arduino software

< <http://arduino.cc> >



# Re-programming the ArduTouch

## Second:

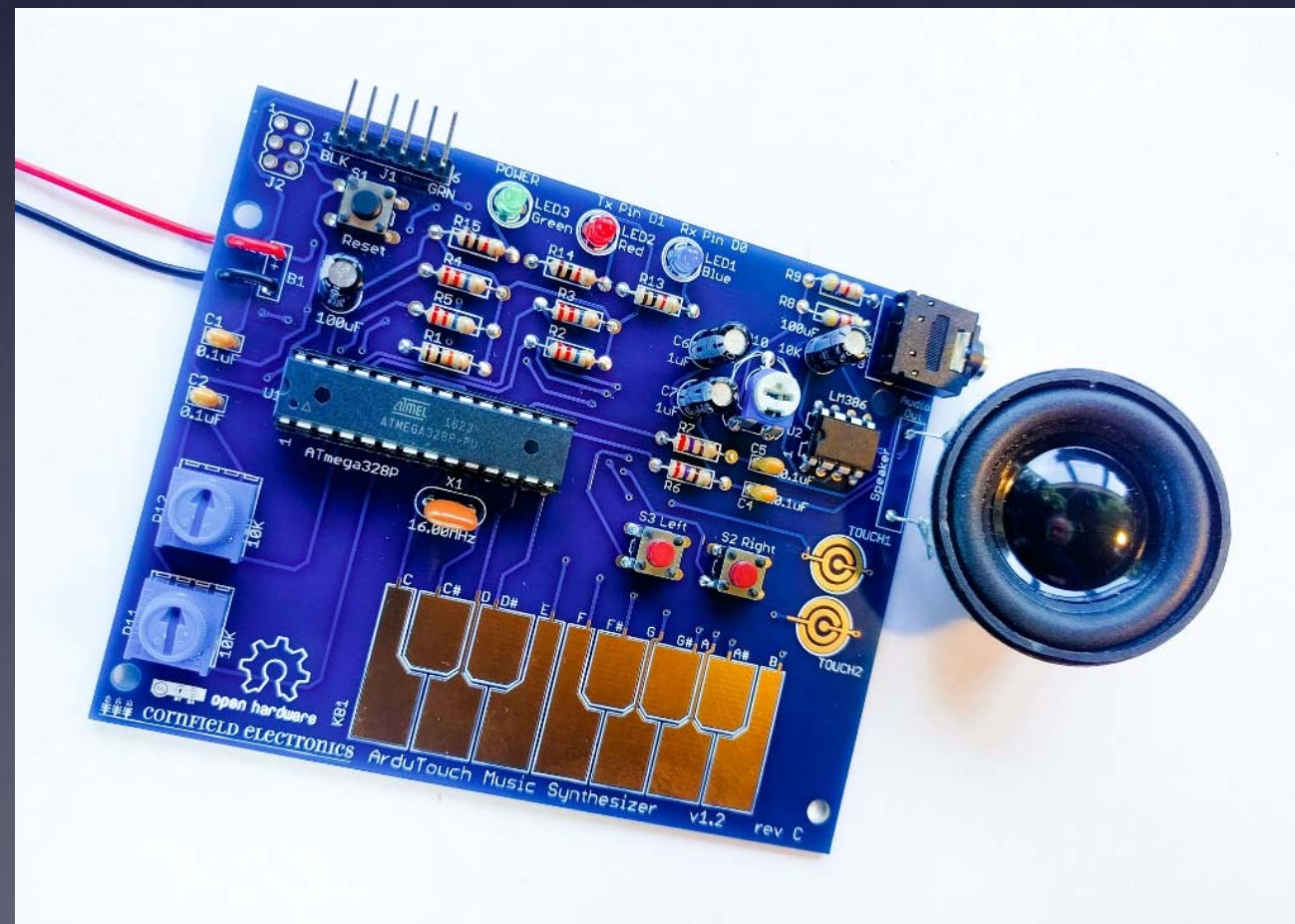
Download and install the ArduTouch Arduino library  
<<http://cornfieldelectronics.com/cfe/projects.php#ardutouch>>

Instructions for installing a .zip library are at:

< <https://www.arduino.cc/en/Guide/Libraries>>

Scroll down till you see the section:

“Importing a .zip Library”



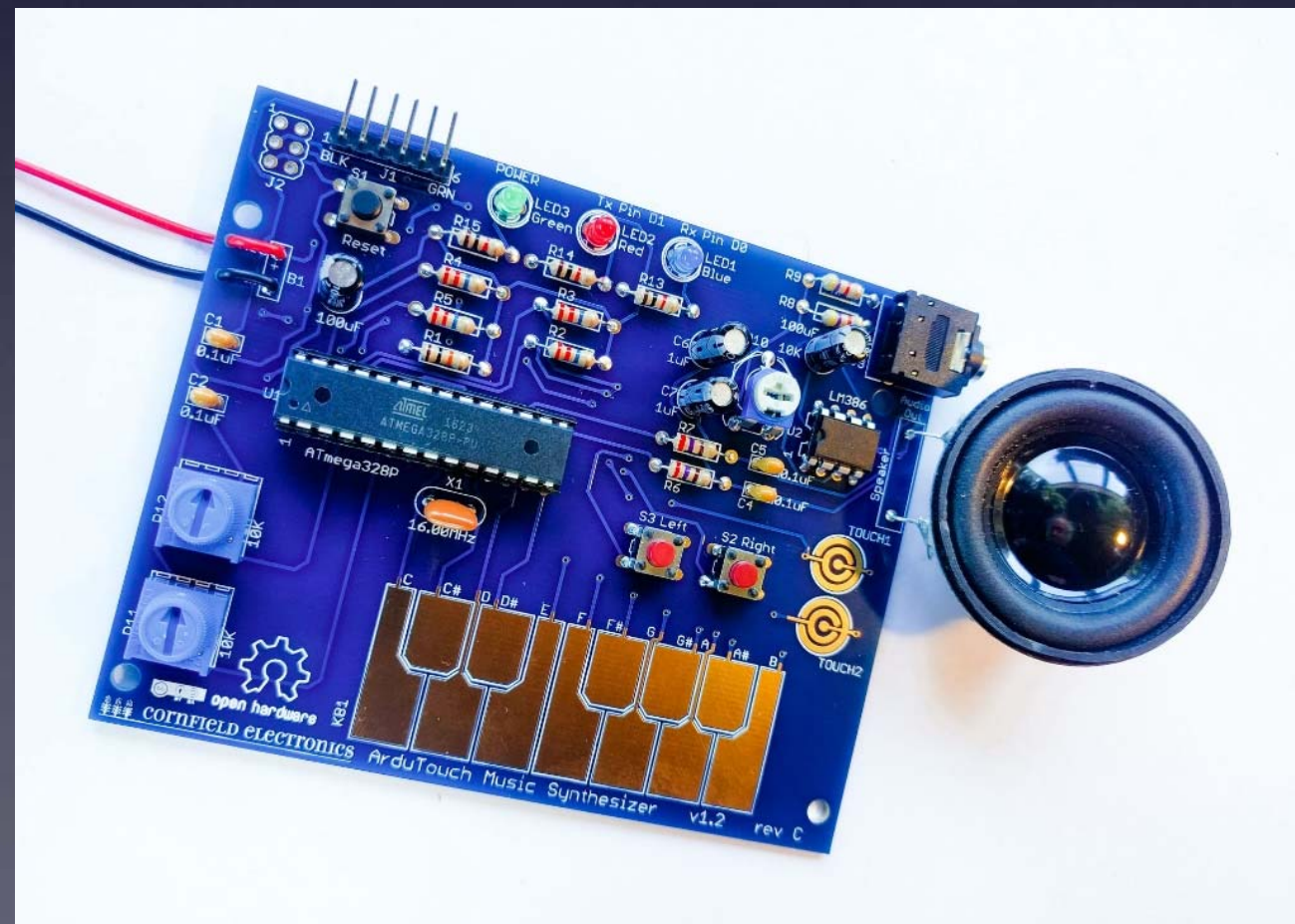
# Re-programming the ArduTouch

## Third:

Download ArduTouch synth sketches

<<http://cornfieldelectronics.com/cfe/projects.php#ardutouch>>

Store them on your computer anywhere you like.

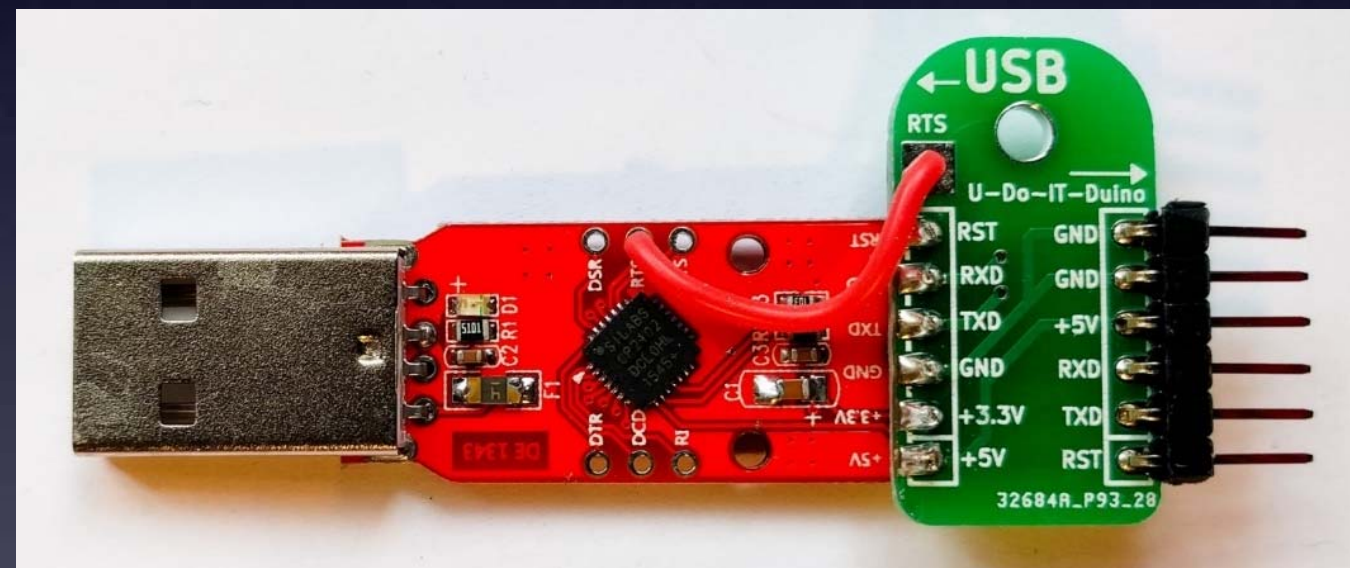


# Connecting your ArduTouch to your computer

## USB-Serial adapter cable

Ones available from Cornfield Electronics look like this:

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

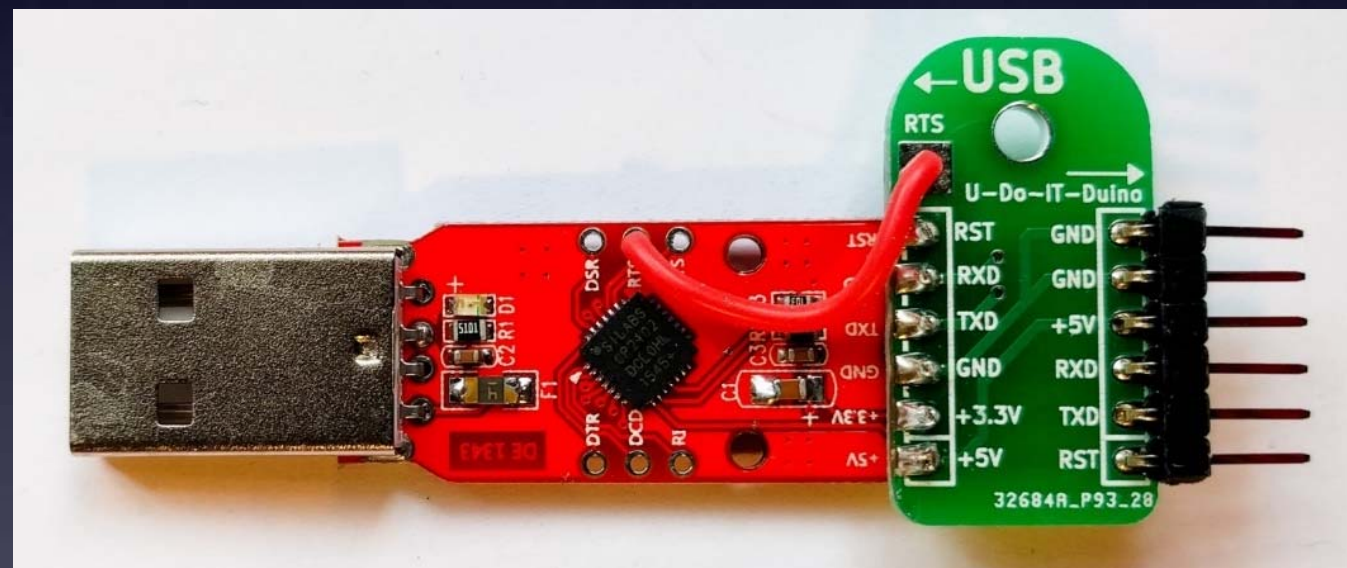


# Connecting your ArduTouch to your computer

## USB-Serial adapter cable

Ones available from Cornfield Electronics look like this:

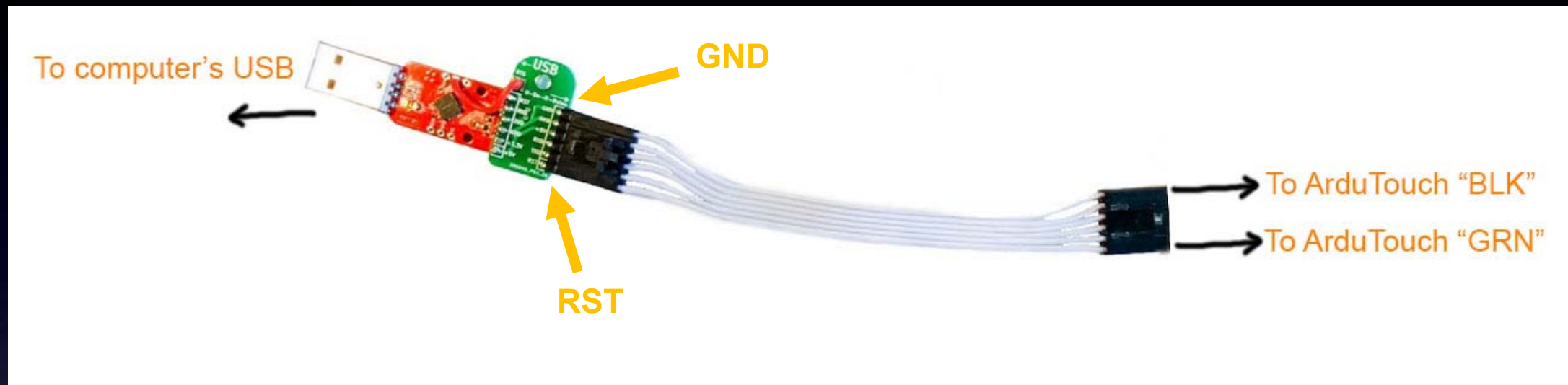
<https://cornfieldelectronics.com/cfe/products/buy.php?productId=usbcable>



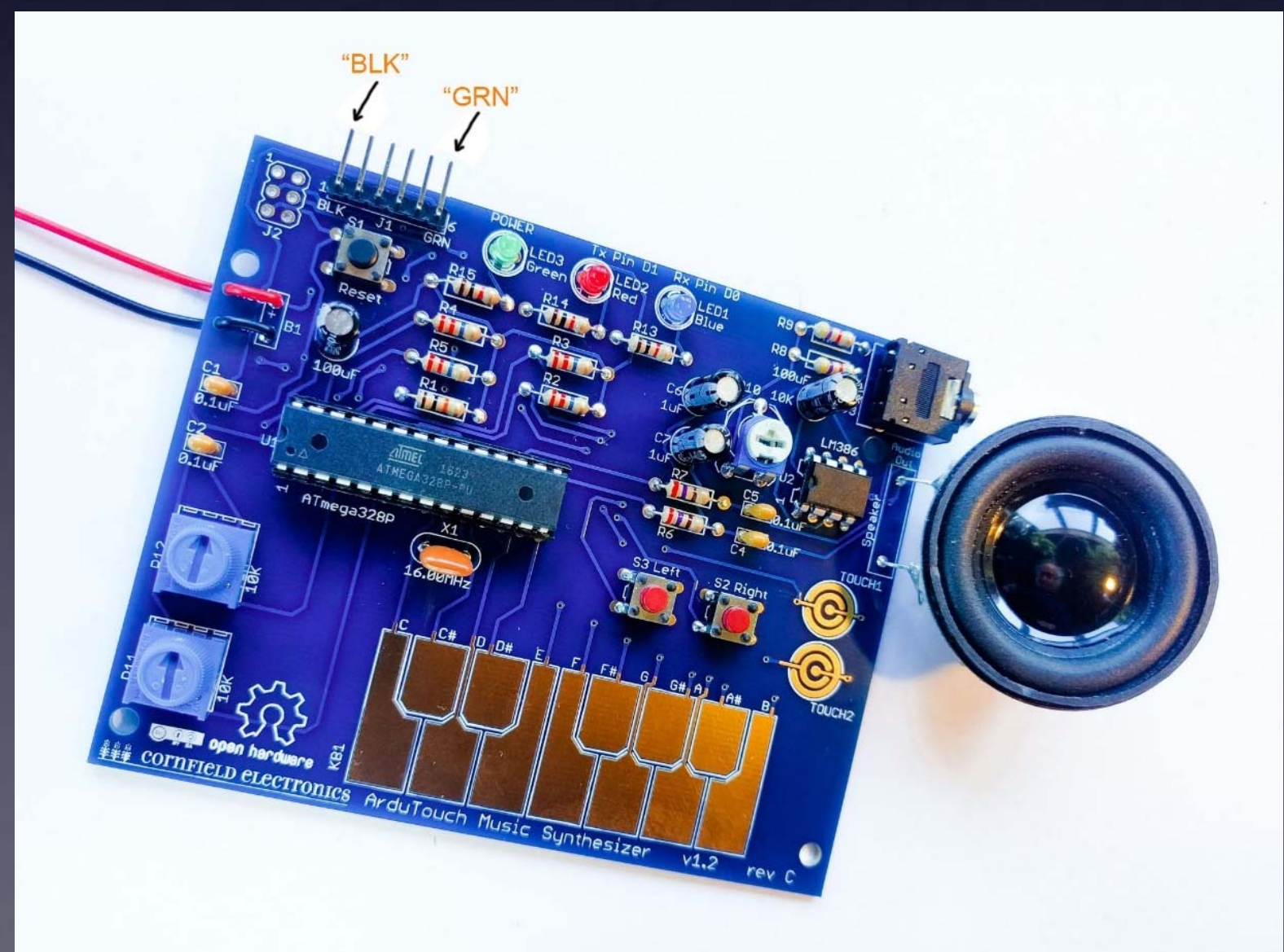
You will need to download and install a driver for your Operating System (Windows, MacOS, or Linux):

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

# Connecting your ArduTouch to your computer

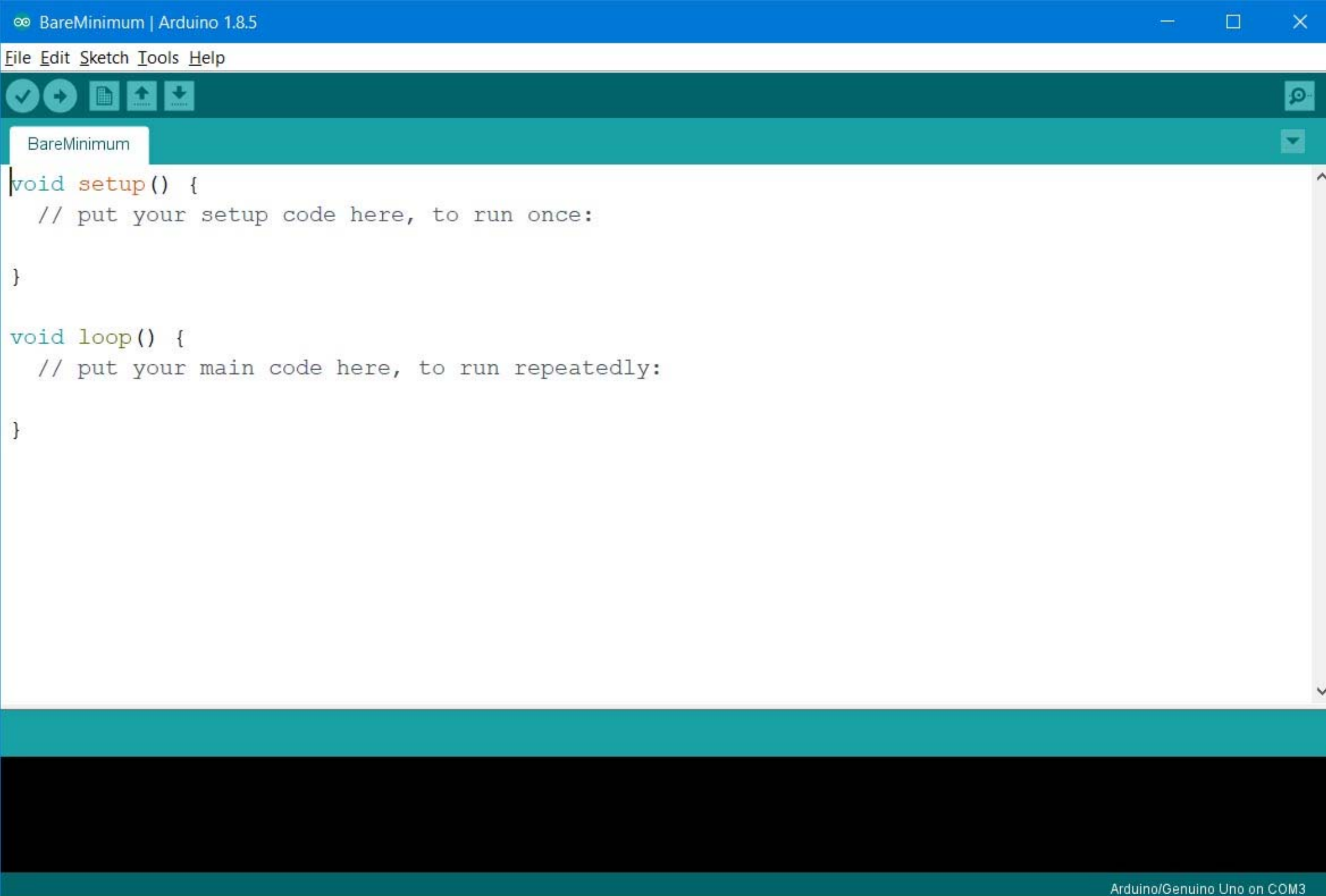


**IMPORTANT:**  
Make sure the  
battery pack on your  
ArduTouch  
is OFF



# Arduino

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



The screenshot shows the Arduino IDE interface. The title bar reads "BareMinimum | Arduino 1.8.5". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checkmark, refresh, file, upload, and download. A tab labeled "BareMinimum" is active. The main text area contains the following code:

```
void setup() {  
  // put your setup code here, to run once:  
  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
  
}
```

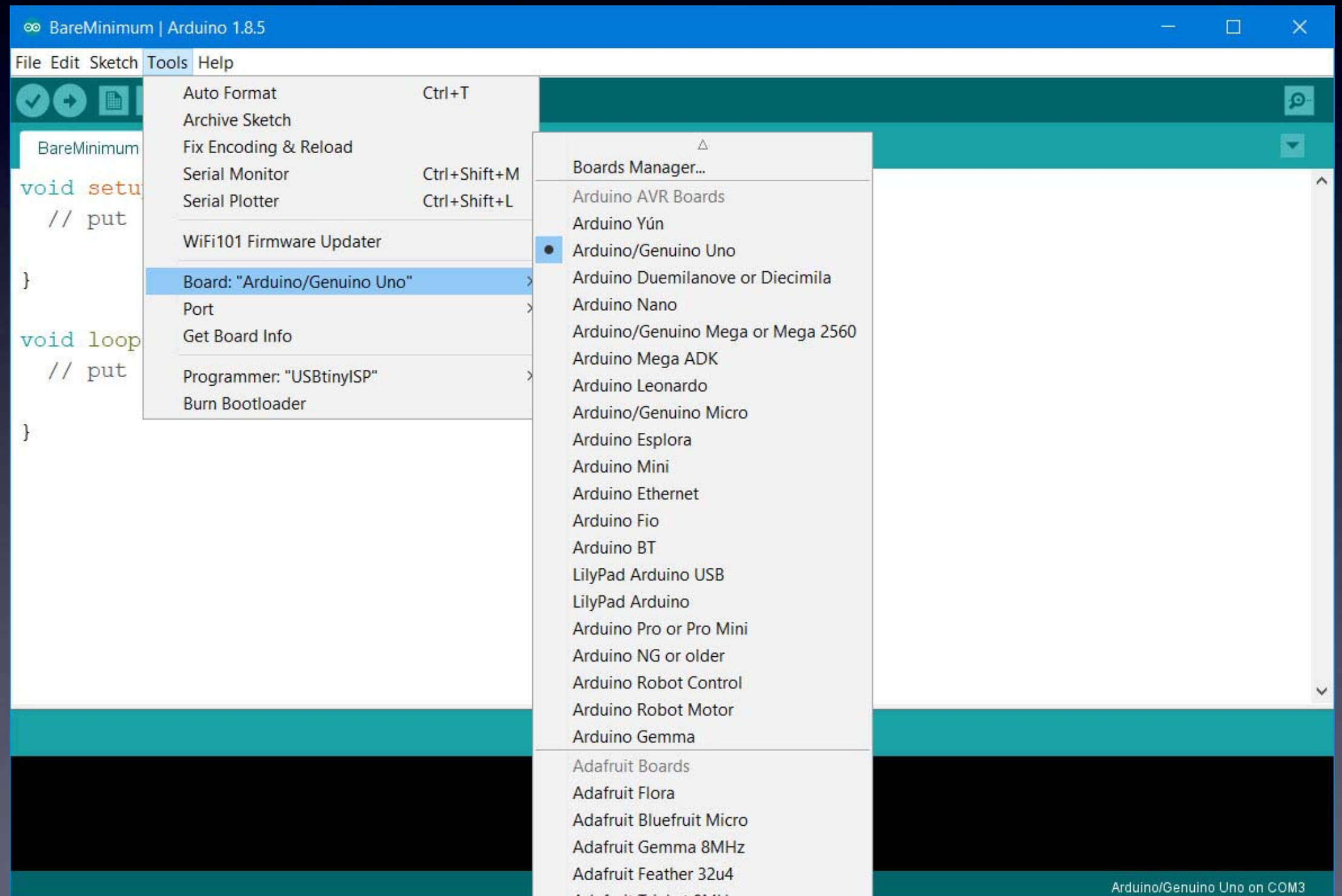
At the bottom right of the IDE, the text "Arduino/Genuino Uno on COM3" is visible.

# Arduino

The first time you start your Arduino software you need to do two things to set things up

(1)  
Choose  
“Genuino Uno”  
as the Board

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





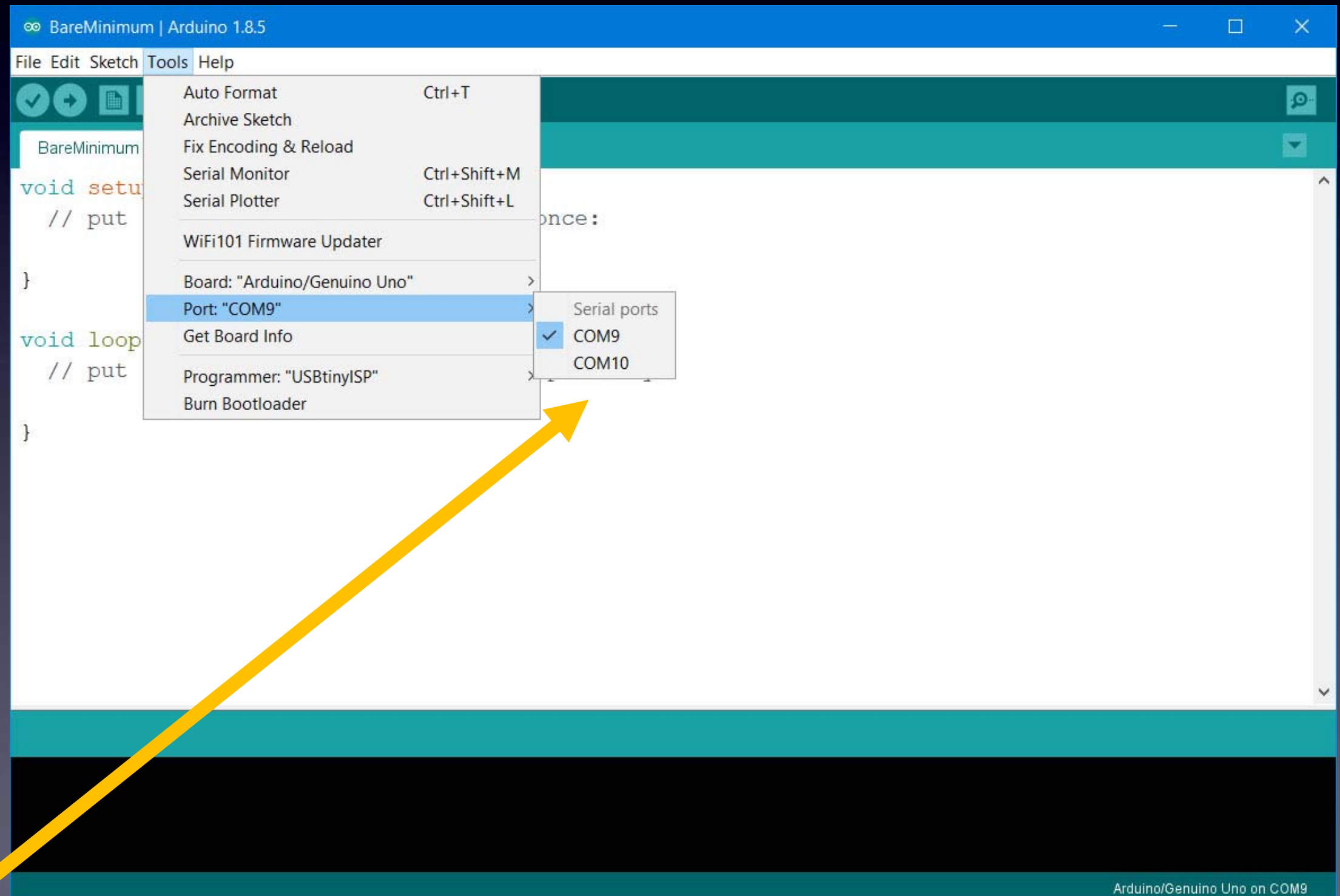
# Arduino

The first time you start your Arduino software you need to do two things to set things up

(2)

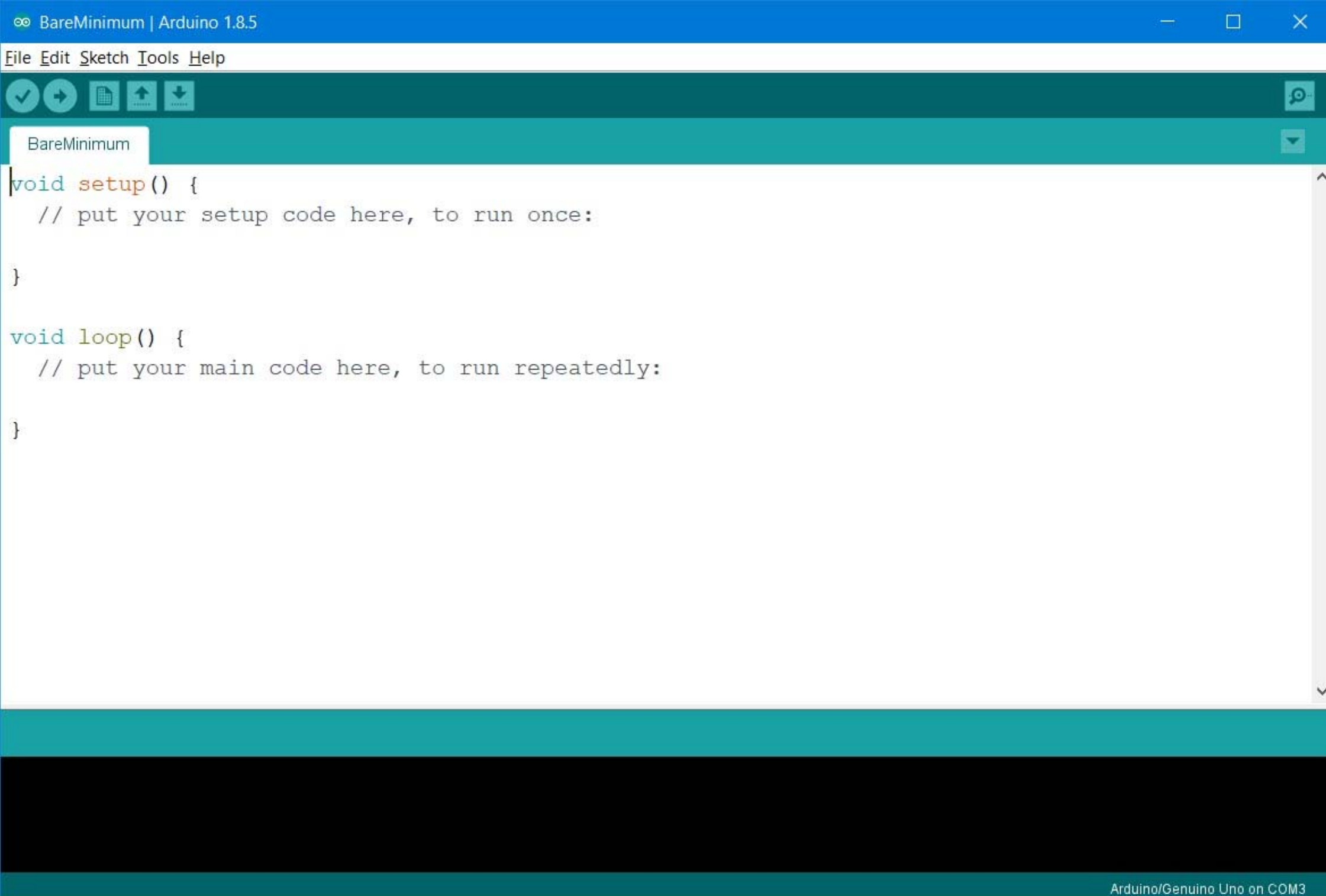
**Choose the Port (this will be different depending on your Operating System)**

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



# Arduino

**Your Arduino software is now ready to program your ArduTouch!**



The screenshot shows the Arduino IDE interface. The title bar reads "BareMinimum | Arduino 1.8.5". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checkmark, refresh, file, upload, and download. A tab labeled "BareMinimum" is active. The main editor area contains the following code:

```
void setup() {  
  // put your setup code here, to run once:  
  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
  
}
```

At the bottom right of the IDE, the status bar displays "Arduino/Genuino Uno on COM3".

# Arduino

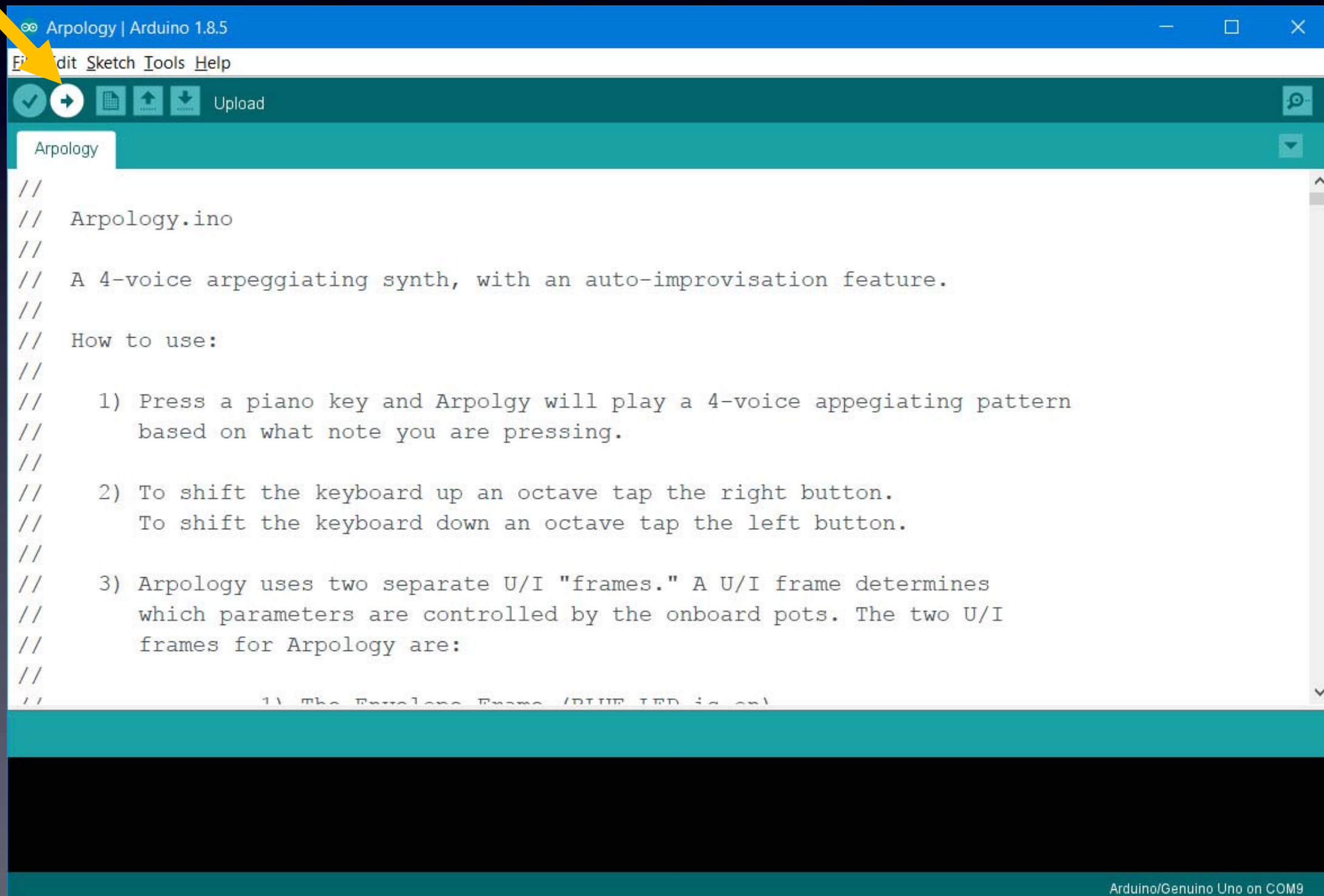
You can open an ArduTouch synth sketch from:  
File → Open...

(I opened “Arpology here)

```
File Edit Sketch Tools Help
Arpology | Arduino 1.8.5
File Edit Sketch Tools Help
Arpology
void setup()
// p
}
//
// Arpology.ino
//
// A 4-voice arpeggiating synth, with an auto-improvisation feature.
void loop()
// p
// 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)
```

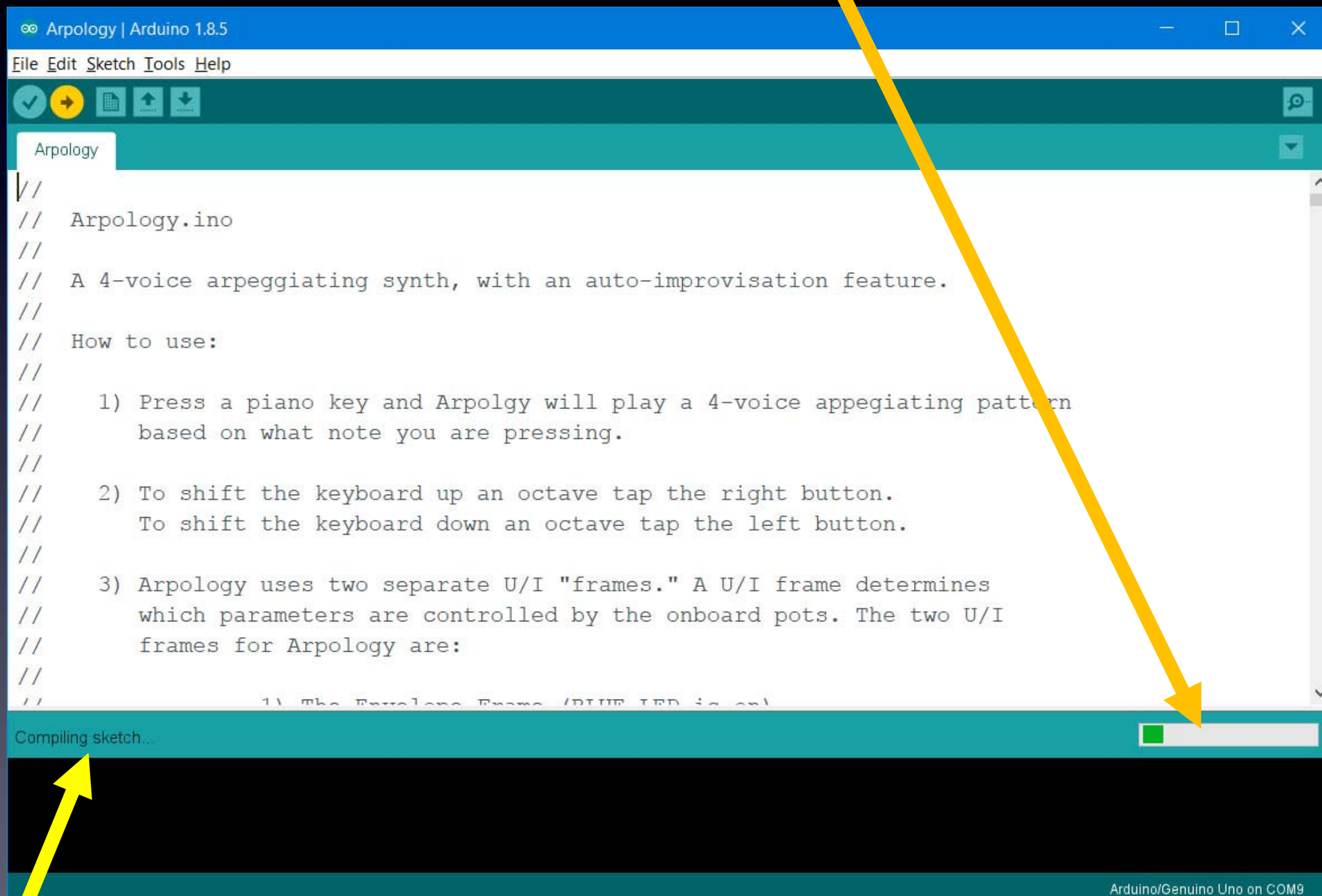
# Arduino

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



# Arduino

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



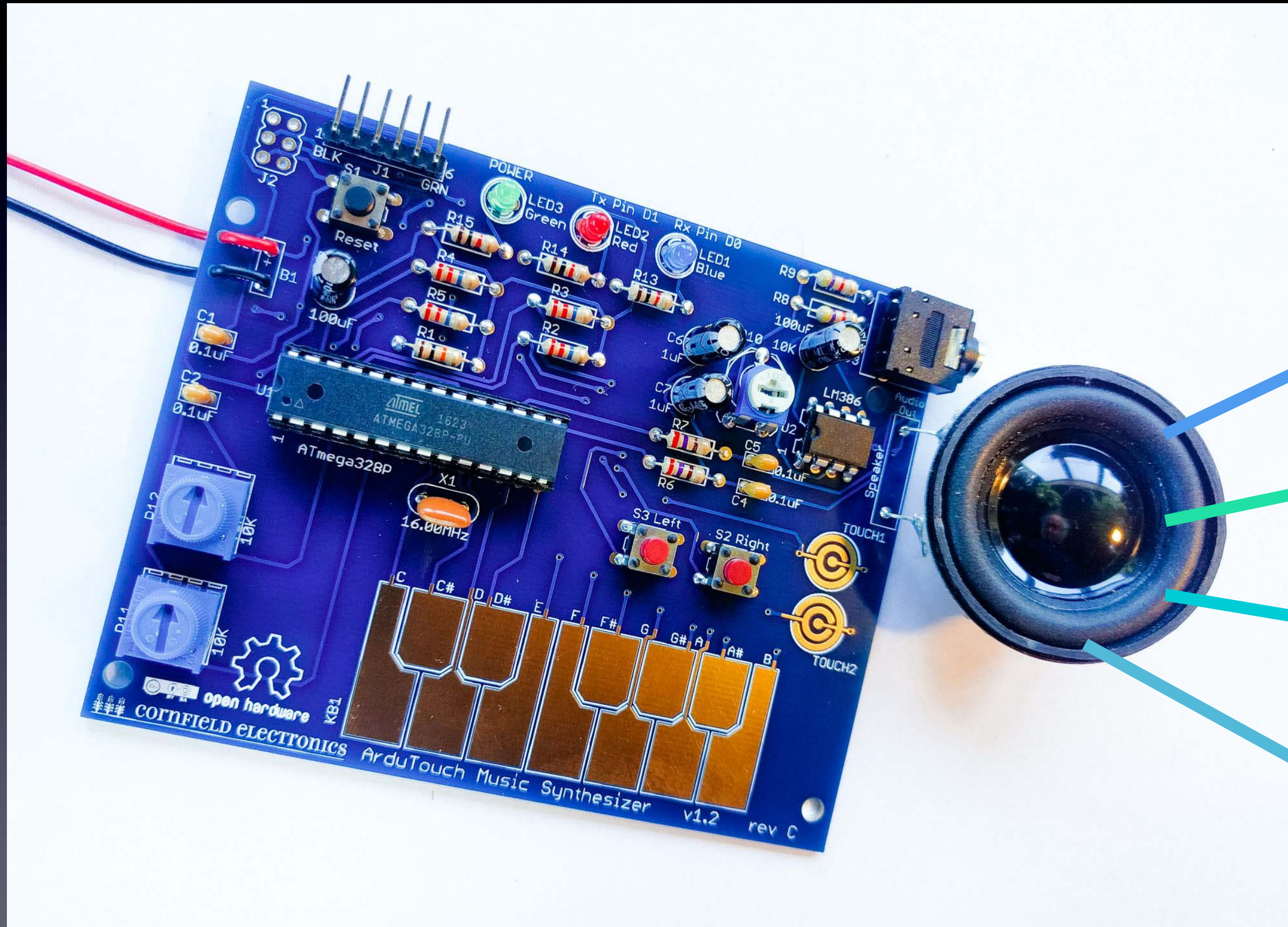
...and when it's completed successfully, it says: "Upload done"

# ArduTouch

**Disconnect your ArduTouch board  
from the USB-Serial cable,  
turn on your battery pack,**

**And...**

# Let's make new noise!



# *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** (1<sup>st</sup> and biggest hardware accelerator)

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

email: [mitch@CornfieldElectronics.com](mailto:mitch@CornfieldElectronics.com)

site: [www.CornfieldElectronics.com](http://www.CornfieldElectronics.com)

twitter: [@maltman23](https://twitter.com/maltman23)

flickr: [maltman23](https://www.flickr.com/photos/maltman23/)

WeChat: [mitchaltman](#)

