Overview

There are lots of ways to make music with the Circuit Playground Express and Microsoft MakeCode. This musical glove lets you control the notes you play, just by moving your hand!

It's inspired by the MI.MU glove (https://adafru.it/Eqs) invented by Imogen Heap, which uses gestures to remotely control a synthesizer, leaving the musician free to move around the stage. The still-experimental gloves have been used in concert (https://adafru.it/Eqt) by Heap and other artists, including Ariana Grande.

I loved Heap's creation so much, I included a DIY version of the MI.MU glove for my book Musical Inventions (https://adafru.it/Equ) a few years ago using sensors, remote control, and synthesizer modules (https://adafru.it/Eqv) from littleBits.

More recently, children's author Helen Leigh (https://adafru.it/Eqw) worked with Heap to make a kit for kids called the MINI.MU (https://adafru.it/Eqx). The MINI.MU uses the BBC micro:bit (https://adafru.it/yEO), and comes with a felt glove for you to sew, adding pockets to hold the board, a speaker, and a battery pack.

This version uses the Circuit Playground Express instead of the micro:bit, which eliminates the need to add a speaker, since there's already one onboard. You don't even need to do any sewing -- just attach the CPX directly to an existing glove and you're good to go!

Parts List -- Electronics

All you need is the Circuit Playground Express, along with the USB cable to program it, and the battery holder to power it when it's not plugged into the computer. All three are included in the Base Kit, or get the parts you need separately. The wires on the battery holder are pretty short, so you may want to add one or more 20-inch-long JST extension cables. That will allow you to carry the battery holder in your pocket (or clip it to an armband, waistband, or shirt sleeve cuff) instead of attaching it directly to the glove.
Materials List -- Household and Crafts Supplies

Other supplies you will need include:

- 1 glove, preferably fingerless, with a tight weave (I used a cotton-spandex "crafter’s comfort glove")
- 4 peel-and-stick Velcro dots
- (optional) armband or strap to hold the battery pack
Assemble the Glove

Although MakeCode lets you test your program on a simulator, you'll want to assemble the Musical Glove beforehand. That way you can make sure the code works the way you think it will when everything’s in motion in three-dimensional space.

Peel-and-stick Velcro dots (or similar hook-and-loop connectors) are an easy way to put everything together. Here's what to do:

Connect the CPX Board to the Glove


Take the matching halves of the dots and press them onto the CPX dots, with the fuzzy/scratchy sides together and the glue side facing out.

Put on the glove. Position the CPX on the back of your hand, with the JST socket for the battery holder pointing towards your wrist. Press the board firmly onto the glove so the glue sticks tight to the fabric.

Now gently detach the CPX, making sure the Velcro dots separate, leaving the bottom dots on the glove. Press them again onto the fabric to bond them firmly. Use care whenever you remove the board to avoid accidentally pulling the dots off the fabric.
Plug in the Battery Holder
Insert batteries into the battery holder. Add one or more JST extension cables if you want to slip the battery holder into a pocket or clip it onto your clothing or an armband. Otherwise, you can use Velcro dots as you did with the CPX and attach it directly to the glove.

Plug the cable into the CPX.

You're ready to start coding the CPX and test it out with your glove!
To make your glove musical, you'll use the CPX like a synthesizer. A synthesizer works by producing an electrical signal and altering it in special ways to create different sounds. Then it sends the signal to a speaker, which translates the changes in voltage into sound wave frequencies you can hear.

Programming the CPX to play a note or musical phrase is super easy with Microsoft MakeCode. This drag-and-drop visual language has music blocks that you can adjust in multiple ways. For this project, you'll start with a single note or series of notes, then vary the sound by changing four things:

- the pitch (how high or low the notes sound),
- the duration of the note (how long or short it is compared to other notes)
- the tempo (how fast or slow they play),
- and the volume (how loud or soft they sound).

If you haven't made your own music before with MakeCode, this page will give you a quick run-down. And if you're new to MakeCode and want to learn more, you can find a full getting-started guide by clicking the button below.

[https://adafruit.it/wWd](https://adafruit.it/wWd)

Using the Music Menu

Open MakeCode online ([https://adafruit.it/wmd](https://adafruit.it/wmd)) or the MakeCode for Adafruit app ([https://adafruit.it/Eql](https://adafruit.it/Eql)) on your computer and click on the Music menu. You'll be combining these orange blocks with other programming blocks to create your code.

MakeCode has a simulator that lets you hear how your code works while you write it. However, it may sound
Creating the Notes and Changing the Duration

To play a repeating note, grab the play tone at Middle C for 1/2 beat block from the MUSIC menu and insert it inside a forever loop block. That's it!

The length of the note is pre-set to one-half beat. To change it, click on that space to open a drop-down menu of choices.

To create a series of notes, just grab another block from the MUSIC menu (or click on and duplicate the first block) and drag it underneath the first block until it clicks into place. Add more notes as needed to play a musical phrase over and over.

When you want a break between notes, use a rest for 1/2 beat block. As with play tone blocks, you can adjust the number of beats the rest lasts.
To mute the music in the MakeCode simulator (which can get annoying while you're trying to fix your code), click the speaker icon (shown in red in the screenshot at the top of the page).
Changing the Pitch

The play tone at block is pre-set to play the note Middle C. To change it, click on the name of the note and a piano keyboard will appear!

Click on any key to set a new note. The name of the new note will appear.

Notice that a number replaces the name of the note on the block itself. You'll learn how to use that number to vary the sound on the next page.

Controlling the Volume

To make the notes louder or softer, use the set volume block.

The volume is pre-set to 128. To change it, click on the number to open up a slider. Drag the slider right or left to increase or decrease the volume.

You can also type a number into the box, over the number 128.

To see how loud that sounds in real life, write a short music program and download it to your CPX. To adjust the default volume for your program, drag a set volume block into an on start block.
Controlling the Tempo

To make the entire musical phrase play faster or slower, use the **set tempo** block. It is pre-set to **120 bpm**, which stands for beats per minute. If you type in a bigger number, the notes will speed up. Type in a smaller number to slow the notes down.

To adjust the tempo at the beginning of the program, use an **on start** block.

To change the tempo while the program is running, use a **change tempo by 20 (bpm)** block. In the example shown here, the entire musical phrase (three notes and a rest) will speed up by 20 beats per minute every time the entire **forever** loop repeats. Type right over the 20 to increase or decrease how much the tempo speeds up. To slow the tempo down, type in a negative number.

Be aware that even though you are changing the tempo, the notes stay the same in relation to each other. The rhythm of short and long notes will stay the same. They will just play faster or slower.

The button below takes you to a preview of the example code above. Listen for a few repetitions and notice how the notes gradually get faster and faster. To play around with the code, click “Edit” and it will open as a new project in MakeCode.

https://adafruit.it/Eqm

On the next page, find out how to vary the pitch, duration, tempo, and volume by changing the position of your musical glove!
There are lots of options for writing simple "synth" programs using MakeCode! Here's an example that uses the `on tilt` input blocks. When you tilt the CPX, it triggers an event that plays notes which go up or down, faster or slower. The button below will open the program in MakeCode so you can test it and play around with it. Then read the instructions to see how it was put together.

https://adafru.it/Eqn

Translating Pitch Into Frequency

When you control musical sounds with movement, you are changing the numbers that represent those sounds: the pitch, the tempo, the duration, and the volume. To change the pitch, you need to replace the names of the notes with their numbers.

As you already saw, every note on the drop-down piano keyboard on a `play tone` shows a number when you press it. This is the frequency of that note. It measures how often a sound wave at that pitch vibrates in one second, and is given in hertz (written Hz).

Middle C has a frequency of 262 Hz. Middle A, five notes higher on the piano keyboard, has a frequency of 440. By using numbers instead of names, the CPX can play a much wider range of frequencies than the keys on the piano keyboard, including tones between standard notes.
Changing Pitch Using Variables
MakeCode includes a *change tempo to* block that lets you adjust the tempo while the program is running.

But to do the same with the pitch, you need a couple extra steps. First, you'll replace the number with a variable. Then you can change the variable to change the number.

Create a Variable for the Pitch
Click on the **VARIABLES** category to open up a box that lets you give the variable a name. Let's call this "Note."

The **VARIABLES** menu now contains blocks with the name **Note**.

To set the value of **Note** when you begin the program, take an **on start** block from the **LOOPS** menu and drag a **set Note to 0** block inside it.

Type in whatever frequency you like. In this example, **Note** starts out as **175**, which is Low F on the piano keyboard.

Use the **on start** block to set the beginning values of the volume and tempo too.
Use the Tilt Blocks to Create Notes

Grab a play tone at block from the MUSIC menu. Then grab a Note oval from the VARIABLES Menu. Drag the Note oval right over the Middle C space and let it snap into place.

Go to the INPUT Menu and drag an on shake block into the workspace. Click on it to open the drop-down menu and choose tilt left.

Insert the play tone at Note for 1/2 beat block inside the on tilt left block. Then get a change Note by 1 block from the VARIABLE menu and add it underneath. Change the number to 30.

Test the code on the simulator. Every time you tilt the board to the left, it should play a note that's a little higher than the last one. (The notes may sound sharp or flat because they are not exactly 30 Hz apart.)

To play a string of notes going up, grab a repeat 4
times} block from the **LOOPS** menu and insert it around the **Note** blocks. You can change the number of times the loop repeats.

It's handy to add some lights to show which way the board is tilted. From the **LIGHT** menu, add a **show ring** block to the stack. Make the lights on the left any color you like by clicking on the color wheel in the center and then on the lights. Turn off the lights on the right by clicking on the grey center and then the lights.

At the very end of the stack, add a **clear** block from the **LIGHT** menu. This turns off all the lights after the code has run.
Add Code for Tilt Right, Up, and Down

Right click on the top of the **on tilt left** stack and choose Duplicate from the list of actions. Then change the input block for the new stack to **on tilt right**.

On the **show ring** block, make the lights on the right a new color and blank out the lights on the left.

Then replace the number in the **change Note by** block to -30. Now, when you tilt the board to the right, it will play a string of notes that get lower.

Add **INPUT** blocks for tilt down and tilt up the same way, but instead of changing the pitch, use the **change tempo by** block to make the tempo go up or down.
More Musical Methods

In addition to using MakeCode’s on tilt and on shake blocks to trigger notes, you can map readings from the CPX’s accelerometer to the frequency of the notes the board is playing. That way, your Musical Glove can play continuous notes that change as you move the board around.

The accelerometer measures the pull of gravity on the board in three directions: along the x axis (right to left), along the y axis (top to bottom), and along the z axis (up and down). Usually the measurements are given in terms of mgs (microgravity). But for this project, it’s a little easier to measure rotation in degrees, like a circle.

To learn more about how an accelerometer works and how to program it in MakeCode, check out the Make It Shake, Rattle, and Roll Guide. (https://adafru.it/Eqo)

How to See What the Accelerometer is Reading

A quick way to see readings from the accelerometer (as well as many of the other sensors on the CPX) is to go to the LIGHT menu and grab a graph 0 block.

Then go the INPUT menu, find the oval for the reading you want to monitor, and drag it over the graph 0 .

When you tilt the board back and forth, the pixels on the simulated board will light up in proportion to the value its reading. The more pixels, the higher the value.

You can also click on the Show Console Simulator button to open a graph and a list of numbers. The graph is marked to show the high and low points, and the numbers show you the readings in real time.

In this case, when you tilt the simulated board as far as it will go, you get high and low readings of around 90 and -90.
If you have the MakeCode app on your computer and plug in your CPX, you can also get live readings from your board!

**Code a Musical Glove That Measures Rotation**

![Code Example](https://learn.adafruit.com/cpx-musical-glove)

This code example plays a tone based on how many degrees the CPX is tilted to the right or left. When the board is flat and facing straight up, that reading is zero. Tilt it all the way to the right at the accelerometer will read -90 degrees. All the way to the left is 90. Depending on how far you want to twist your wrist beyond that, you can set the range from about 130 to 150 (and -130 to -150).

https://adafruit.it/Eqp

https://adafruit.it/Eqp
Write the Rotation Code

Open the **INPUT** menu and scroll down to the bottom to grab the `rotation (°) pitch` block. The "pitch" is the measurement side to side. "Roll" measures tilt from top to bottom.

Next, go to the **MATH** menu and grab the `map 0 from low 0 high 1023 to low 0 high 4` block, all the way at the bottom.
Drag the **rotation** block over the first 0 until it snaps into place. Then type in numbers over the remaining spaces. This will "map" or translate the rotation range you want to read to the frequencies you want the CPX to play.

The first two numbers represent the low and high end of the rotation you want to measure, in terms of degrees. In this example, -130 was typed over the next 0, and 130 over the 1023.

The next two numbers are the low and high end of the notes you want to play, measured in hertz. Choose whatever notes sound good to you. For this example, 27 was typed over the third 0, and 1500 over the 4.

Next, put a **play tone** block inside a **forever** block and drag the **map** block you just created over the **Middle C**.

The final step is to add any additional blocks you like to adjust things like volume and tempo.

---

**Add to Your Musical Glove Program**
You can build on the rotation code by creating a phrase of two or more notes. If you start with the rotation value and multiply it by a whole number or a decimal (using blocks from the MATH menu), the intervals between the notes will stay the same as the string of notes get higher and lower. Musical intervals like a fifth (0.2) and a third (0.3) will sound good. Multiplying a note by one half (0.5) or doubling it will give you the same note, just an octave higher or lower.

To see how that works, click on the green button and listen to the simulation of a program using intervals.

You can try all kinds of combinations of input to create different sounds. (You can even use readings from other sensors, such as the light sensor.) Play around with them to make something that sounds good to you!