Morse Code, Android, Assistive Tech

by Limor “Ladyada” Fried
(ladyada@alum.mit.edu) and
Phillip Torrone (fill@2600.com)
-. . - . . - / . . - .

In the 1800s, all the cool kids were texting each other. No, it wasn’t SMS, iMessage, or SnapChat. It was called Morse code (named for Samuel Morse, inventor of the telegraph). Morse code encodes letters and numbers into “dots” and “dashes.” A lot of people learn “SOS” (which is * * * - - - * * *) from watching movies, imagining one day they’ll need to tap the sequence to escape from somewhere.

Later on, amateur radio operators (HAM) used Morse code, even before voice radio was available. For decades, to get an amateur radio license, the operator needed to demonstrate they could do at least five words per minute in Morse code and, for the highest (amateur expert class), 20 words per minute. The FCC, which governs all of this, reduced the Morse code requirements in 2000 for the words per minute, then eventually in 2007 completely eliminated the Morse code requirements for all amateur radio licenses. The reason for the change was it “...eliminates an unnecessary regulatory burden that may discourage current amateur radio operators from advancing their skills and participating more fully in the benefits of amateur radio.” Anyone who still wants to use low-power or low-speed communication can use a computer or microcontroller digital Morse code converter instead of tapping by hand.

So here we are, 2018, and while Morse code will always have some uses, it certainly is not as popular as it was in the past. One of those uses is with Assistive Technology (AT).

AT is something we’ll all need. We’re getting older, our meat bodies are sticking around longer but fail in one way or another. From accidents to being born with special needs, there are many situations where communication may be a challenge for some or all of us one day. So being able to communicate without voice or wide range movements will be essential.

In the maker and hacker community, one great use of our skills is helping others in the assistive technology community who need smart solutions for communication and access to technology with limited mobility. With low-cost microcontrollers, “internet of things” devices, and being able to modify many off-the-shelf electronics to work better for the disabled community, it’s never been a better time to help each other.

One of the cool “tricks” of some of the modern low-cost microcontroller platforms (Arduino, Circuit Playground, Feather, Makey Makey) is they can act as “USB Human Interface Devices” or HID. This means they can be used as keyboards and mice and still be programmable devices. This is handy for making that interactive art project that plays music when you tap on Jell-O cubes. Or, you can use the same interface to make a device that is not a keyboard into a keyboard for someone with limited mobility. For example, some people may only have the ability to tap
a single button, or “sip and puff” via a tube. Each person will have special interface needs, but we can convert that special interface into standard USB HID. This is where combining a modern device like an Android phone, a HID-capable microcontroller board, and Morse code can help.

Android recently added GBoard, a special keyboard that can be programmed to turn custom inputs into keycodes that any app can use. (support.google.com/accessibility/android/answer/901181?hl=en) Google worked with Tania, who uses Morse code to communicate with a custom-made device to bring it to more people with similar needs. (There’s a video about Tania’s story at youtube.com/QM4QHc6.)

It’s free, so start by searching and installing the “GBoard” app. After following the instructions for running Gboard, you can start Morse codin’ away by tapping two pads on your touchscreen. But what if you want to add a hardware device with a real physical input like a pressure sensor for sip-and-puff, or a large easy-to-press arcade button, or even a muscle sensor?

Thanks to the universal nature of HID, it’s easy to convert any sensor input into dots and dashes using a microcontroller. What’s particularly nice is instead of having to do Morse code detection and decoding on the hardware peripheral, we just need to emit dots and dashes. This makes the hardware easier to build, adapt, and then, of course, the phone can be programmed for specialized shortcuts as desired.

Just about any microcontroller development board with native USB can handle HID output. Note that microcontrollers with USB-to-Serial converters (FTDI/CP210x) cannot do HID, so find a chip that can! We recommend Circuit Playground Express (adafruit.com/circuitplayground), our low cost and fully open source all-in-one dev board. You can use Arduino, MakeCode block based programming, or even CircuitPython. (Fun fact: The developer who added HID support to CircuitPython did so specifically because they were building an AT device for a friend!) A wearable-friendly board works well because the sewable pads can also be grabbed by alligator clips.

You may not even need external buttons or switches. Here’s some example CircuitPython code for using capacitive touch pads instead of mechanical switches:

```python
from adafruit_circuitplayground import express

# You can adjust this to get the level of sensitivity you want.
cpx = express.circuitplayground

cpx.adjust_touch_threshold(100)

while True:
    if cpx.touch_A4:
        kbd.send(Keycode.PERIOD)
        while cpx.touch_A4:
            pass
    elif cpx.touch_A3:
        kbd.send(Keycode.MINUS)
        while cpx.touch_A3:
            pass
```

More code and wiring diagrams are available at learn.adafruit.com/android-gboard-morse-code-at-with-circuitplayground-express.

Keeping the hardware simple as seen above is key to making reliable AT devices - don’t think that more is more! AT users will appreciate something that works every time, and that can be built upon. Of course, you can also add more inputs that have hot-key like commands, such as opening up an app or, if there’s a single input, looking for how long the button has been pressed to determine whether to send a “.” or a “-“.

If you have basic electronics, programming, or 3D printing skills, the AT community would love to get your skills into use. Many parents and teachers know exactly what they need, but don’t know how to build it. Visit atmakers.org or look for a local AT group to volunteer your hacking to help others.

Good night and good luck.