USB Keyboard-Mouse with Teensy

By Christian, updated on 18 July 2018

Hey all! By now you should have heard about all the wonderful **Teensy Boards** produced by Paul Stoffregen and the PJRC team but, in case you haven't, they're a collection of high-speed, ARM-based microcontrollers that have the ability to act as a variety of native USB devices. If you'd like to learn more about the Teensy platform, there are plenty of **tutorials** where it's explained in more depth.

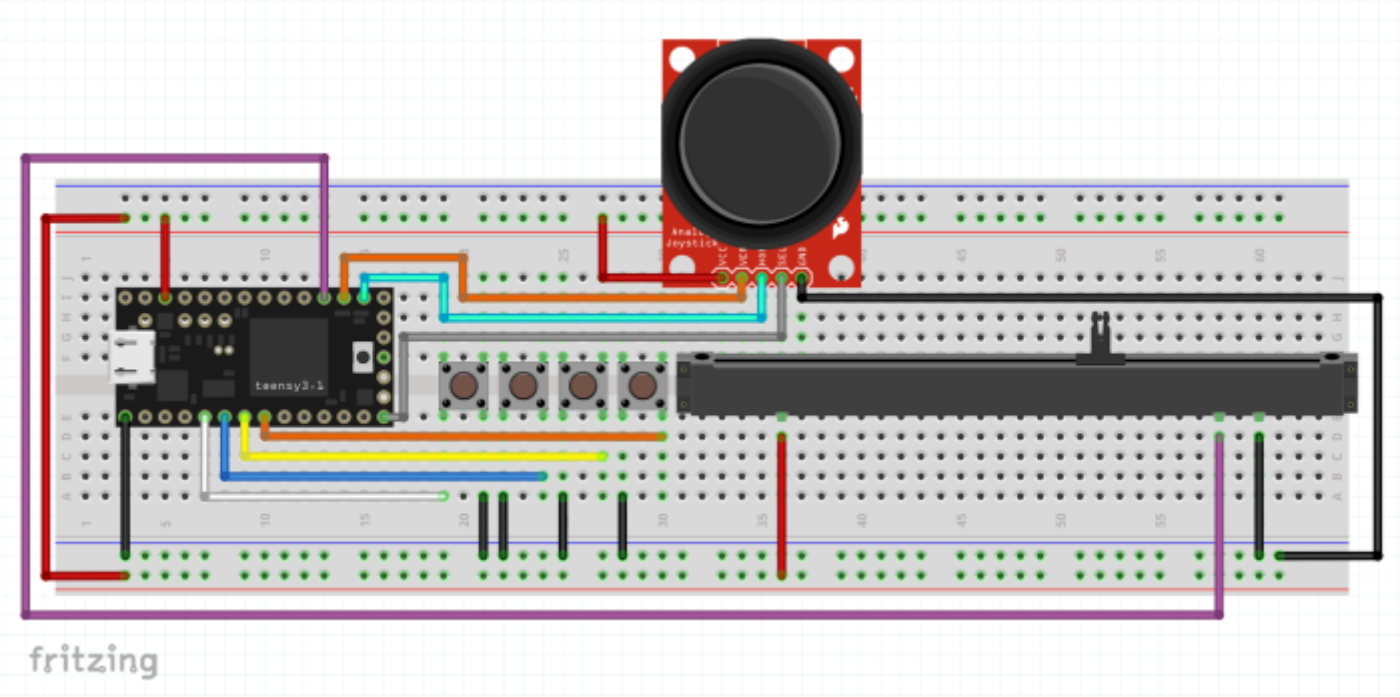
This tutorial uses the Teensy's native USB functionality to act as both a Keyboard and Mouse. This is a great project idea for anyone who works with applications or programs that require excessive hotkey or shortcut use. The code in this tutorial is designed to be used with the 3D design program **Autodesk Inventor Professional 2018** but can be modified to fit any key/mouse input. If you'd like to see an extreme of this application, check out this **Emoji Keyboard**.

THE SETUP

For this project, I'll be using the **Teensy 3.2** due to it's smaller size and price. This Teensy has a whopping 33 digital pins available on-chip, allowing us to connect a huge amount of inputs should we require them. This project, however, is quite simple; we require:

- 4 Non-Latching **Pushbuttons**
- A **Slide - Potentiometer**
- A **Joystick**.

These components have enough versatility to enact 8 different, user-set commands as well as move the cursor around via the Joystick. If you'd like, you can encase the assembled project within an enclosure or keep them on a prototyping breadboard for frequent modifications and input additions. The assembly of the pieces above should look like the photo below (disregard the Teensy Model Name)..



For this program to work, we must install the TeensyDuino Application from the PJRC website. The installation process, along with a walkthrough of the changes to the Arduino IDE, can be found in the Using Teensy with Arduino IDE tutorial.

CODE

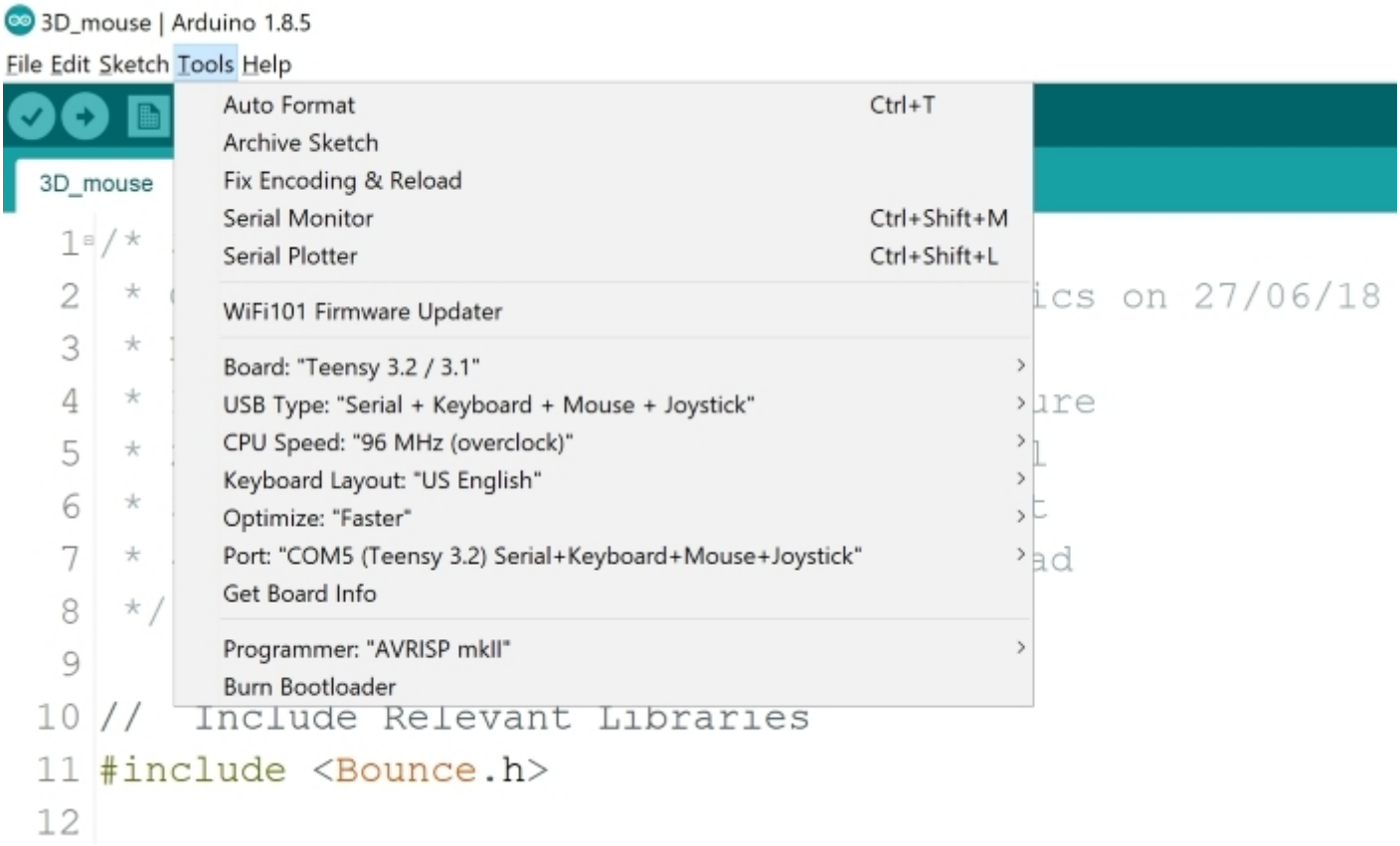
As mentioned previously, the example code for this tutorial is centred around a 3D design program but can be modified to suit any application or program which incorporates hotkeys or shortcuts. The .ino file can be found on **GitHub** and also appears below.

```
  if(buttonArray[i].fallingEdge()){
    //  Check to see if the shiftState is NOT active
    if(!shiftState){
      //  Apply the normal state keybinds
      Keyboard.set_modifier(KEYBIND_MOD_SHIFT_P[i]);
      Keyboard.set_key1(KEYBIND_SHIFT_P[i]);
      Keyboard.send_now();
      Serial.println(KEYBIND_MOD_SHIFT_P[i]);
      Serial.println(KEYBIND_SHIFT_P[i]);
      }else{
      //  Apply the shift state keybinds
      Keyboard.set_modifier(KEYBIND_MOD_SHIFT_NP[i]);
      Keyboard.set_key1(KEYBIND_SHIFT_NP[i]);
      Keyboard.send_now();
```

UPLOADING AND FIRST RUN

As mentioned underneath the schematic, the TeensyDuino application must be installed before the code can be uploaded to the board. From there, the board's preferences must be configured

computer's setup. Once that's done, hit the upload button and watch as your Teensy becomes a handy shortcut-keyboard/mouse! The shortcuts are specified in the comments at the top of the program and are suited to work with Autodesk Inventor Professional 2018 as mentioned previously.



PERSONALISING YOUR TEENSY KEYBOARD/MOUSE

Now that you've had some fun inspecting and editing your 3D models with your Teensy powered Keyboard-Mouse, it's time to modify the code to suit your personality. If you find yourself using a shortcut more frequently than you'd like, you can edit the **KEYBIND_SHIFT_P[BUTTON_NUM]** array as well as the **KEYBIND_MOD_SHIFT_P[BUTTON_NUM]** array and replace them with the new shortcut key and modifier respectively. A list of each of these keys and modifiers can be found on the **PJRC website**. If you have any questions about building this handy project, don't hesitate to consult our **forum**.

Category: **Teensy**

Tags: **keyboard mouse projects teensy**

Have a question? Ask the Author of this guide today!

2 replies

John102761 Nov '19

Hi Christian,
Thanks for the post.
I'm looking for a way to use a morse paddle (or straight morse key) to emulate a full ascii keyboard. In the context of 'assistive' technology, I can send morse code at 30 WPM which is much faster than I can type on a PC keyboard. Perhaps something like: morse key > micro-controller > USB Keyboard.
Any guidance will be much appreciated.
Regards,
John Loftus
VK4CT

1 reply

Clinton ▶ John102761 Nov '19

Hi John,

This tutorial is a great place to start.
For what you need, you would only need to have one button.
The trick is going to come in how you will convert your button presses into keyboard characters.
Once you have a button wired in I would record yourself "Typing" some morse code with a for loop that records the presses you make so you can find the timing for your dots and dashes.

Alternatively, you could have a separate button for dashes and dots.