Fingerless Xbox Controller

Brooke Snow
FINGERLESS XBOX CONTROLLER

Brooke Snow

2023
System Design

The Fingerless Xbox Controller (FXC) aims to make gaming more accessible to those who lack the full use of all their fingers by providing a better layout for the inputs on an ordinary Xbox controller. The FXC communicates with a computer that operates on Windows OS via a USB 2.0 and needs a USB-C cable to make that connection. That signal goes directly to the MicroMod, and the 5V supply from the USB gets run through a linear regulator to become a regulated 3.3 volts. That supplies power to the MicroMod and all of the buttons and joysticks. The buttons are attached to an internal pullup resistor in the MicroMod, as are R3 and L3 in the joysticks. Pushing a button or joystick pulls the GPIO (or ADC in the case of the joysticks) to ground, giving an active low signal. The joysticks x and y axes are based off potentiometers in the joysticks that act as variable voltage dividers and send that voltage to the ADC of the MicroMod. The MicroMod takes these signals from the user inputs and translates them into Xinput standard USB communication, which then gets sent to the computer to process. All these inputs are laid out on one flat surface so that you do not need to hold the FXC while playing with it.
PCB Design

Figure 1: Top View of Entire PCB, Top copper prominent

Fingerless Xbox Controller
Design by: Brooke Snow
Iteration 2
03/06/23
Figure 2: Flipped view of Entire PCB, Bottom copper prominent
Figure 3: Zoomed in view of USB-C header, fuse, cut-trace, and USB-C resistors wiring.

Figure 4: Zoomed in view of M.2 Mount for MicroMod
Figure 5: Zoomed in view of LDO and surrounding circuitry to set correct output voltage.

Figure 6: Zoomed in view of example buttons, SWD, and through-hole setup for M.2 Mount.
Figure 7: Zoomed in view of Reset button, other buttons nearby, and joystick.

Figure 8: Zoomed in view of boot button as well as vdiv3 signal voltage divider
Bill of Materials

<table>
<thead>
<tr>
<th>Component</th>
<th>Designator</th>
<th>Description</th>
<th>Footprint</th>
<th>Manufacturer</th>
<th>Quantity</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solder J1, R1 and R2. Connect a USB-C cable to J1 and measure the voltage across TP 6 and TP10 to verify it is 5V.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder F1, IC1, R3, D1, C1, C2, C3, C4, C5 and C6. With USB-C connected to J1, measure the voltage across TP 9 and TP 10 to verify it is 3.3V.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder S1 and S2. While S1 is pressed and USB-C connected to J1, measure voltage across TP12 and TP10 to verify it is 0V. While 21 is pressed, measure voltage across TP13 and TP10 to verify it is 0V.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder J2 and H1. Insert MicroMod module. With USB-C connected to J1 and the other end of the cable connected to a PC, open Arduino (with Teensyduino installed) and upload a blank sketch. Press S2 when sketch is processed to “Boot” the Micro Mod and verify communication with the MicroMod. Computer should respond to the boot signal through Teensyduino’s interface as well as loading code onto MicroMod.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder J3. With USB-C connected to J1, use a Segger J-Link to connect to a PC from J3. Open J-Link Commander and follow steps to validate successful connection to J-Link.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder R4 and R5. Measure voltage across TP14 and TP10 to verify it is 1.67V.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder S3. With USB-C connected to J1, while pressing S3, measure voltage across TP15 and TP10 to verify it is 0V. Release S3 and verify that it is no longer 0V.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder S4. With USB-C connected to J1, while pressing S4, measure voltage across TP16 and TP10 to verify it is 0V. Release S4 and verify that it is no longer 0V.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder S5. With USB-C connected to J1, while pressing S5, measure voltage across TP17 and TP10 to verify it is 0V. Release S5 and verify that it is no longer 0V.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solder S6. With USB-C connected to J1, while pressing S6, measure voltage across TP18 and TP10 to verify it is 0V. Release S6 and verify that it is no longer 0V.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
11. Solder S7. With USB-C connected to J1, while pressing S7, measure voltage across TP19 and TP10 to verify it is 0V. Release S7 and verify that it is no longer 0V.
12. Solder S8. With USB-C connected to J1, while pressing S8, measure voltage across TP20 and TP10 to verify it is 0V. Release S8 and verify that it is no longer 0V.
13. Solder S9. With USB-C connected to J1, while pressing S9, measure voltage across TP21 and TP10 to verify it is 0V. Release S9 and verify that it is no longer 0V.
14. Solder S10. With USB-C connected to J1, while pressing S10, measure voltage across TP24 and TP10 to verify it is 0V. Release S10 and verify that it is no longer 0V.
15. Solder S11. With USB-C connected to J1, while pressing S11, measure voltage across TP25 and TP10 to verify it is 0V. Release S11 and verify that it is no longer 0V.
16. Solder S12. With USB-C connected to J1, while pressing S12, measure voltage across TP26 and TP10 to verify it is 0V. Release S12 and verify that it is no longer 0V.
17. Solder S13. With USB-C connected to J1, while pressing S13, measure voltage across TP27 and TP10 to verify it is 0V. Release S13 and verify that it is no longer 0V.
18. Solder S14. With USB-C connected to J1, while pressing S14, measure voltage across TP28 and TP10 to verify it is 0V. Release S14 and verify that it is no longer 0V.
19. Solder S15. With USB-C connected to J1, while pressing S15, measure voltage across TP29 and TP10 to verify it is 0V. Release S15 and verify that it is no longer 0V.
20. Solder S16. With USB-C connected to J1, while pressing S16, measure voltage across TP30 and TP10 to verify it is 0V. Release S16 and verify that it is no longer 0V.
21. Solder U1. With USB-C connected to J1, while pressing the joystick in, measure voltage across TP23 and TP10 to verify it is 0V. Release the joystick and verify that it is no longer 0V.
22. Measure voltage between TP8 and TP10. While gathering this measurement, move the joystick all the way to the left, then all the way to the right to verify that one direction results in 3.3V and the other results in 0V. Repeat this step with the vertical axis and the voltage between TP22 and TP10. The direction which the voltage is at is max or min is irrelevant as that can be fixed in code.
23. Solder U2. With USB-C connected to J1, while pressing the joystick in, measure voltage across TP32 and TP10 to verify it is 0V. Release the joystick and verify that it is no longer 0V.
24. Measure voltage between TP4 and TP10. While gathering this measurement, move the joystick all the way to the left, then all the way to the right to verify that one direction results in 3.3V and the other results in 0V. Repeat this step with the vertical axis and the voltage between TP32 and TP10. The direction which the voltage is at is max or min is irrelevant as that can be fixed in code.
Software Dataflow Diagram

Verification of Functionality
The following video contains verification that the Fingerless Xbox Controller inputs communicate with the computer and therefore work as intended:
https://youtu.be/2p_fDrv6Go8

Presentation Poster
The following page contains a copy of my final poster used to present to colleagues and industry experts.

Presentation Video
In this video I give a presentation very similar to what I gave during demo day, noise included. I captioned the video, so feel free to turn on closed captioning if necessary.
YouTube Video of Demo Day: https://youtu.be/Q-L18eyKbQ0
Fingerless Xbox Controller

Objectives
In designing the Fingerless Xbox Controller, I wanted to create a device that:
1) is easier to use for someone who had hands that are not the same shape as most.
2) is satisfying to use.
3) lets the user know if they pressed a button regardless of what the game is showing them.
4) allows the user to play games on their Windows OS device.

Summary of Results
The Fingerless Xbox Controller transmits data to the computer every 126 microseconds, or almost 8000 times a second, as seen being measured below. This makes the response time to user input unnoticeable to the user.

The final board (seen in button diagram) is a 10 cm x 10 cm (3.9 in x 3.9in) square. This board is too small for the requirements. While someone with a mild amputation (see Gary’s hand in the motivation section) may have an easier time using this controller over a traditional controller, someone with Symbrachydactyly (See Nicki’s hand in the motivation section) may have more difficulty navigating around the joysticks and close buttons.

Conclusion
The Fingerless Xbox Controller does most of what it aims to. It provides a way for those who do not have normal use of their fingers to play games and reach all of the inputs. However, after some testing, I’ve determined that, in order for all the buttons to be laid out with enough space to press and maneuver with ease, the joysticks should be off board via jumpers.

Abstract
The Fingerless Xbox Controller (FXC) aims to make gaming more accessible to those who lack the full use of all their fingers by providing a better layout for the inputs on an ordinary Xbox controller. The FXC communicates with a computer that operates on Windows OS via a USB. That signal goes directly to the MicroMod, and the 5V supply from the USB gets run through a linear regulator to become a regulated 3.3 volts. That supplies power to the MicroMod and all of the buttons and joysticks. Pushing a button or joystick pulls the GPIO (or ADC in the case of the joysticks) to ground, giving an active low signal. The joysticks x and y axes are based off potentiometers in the joysticks that act as variable voltage dividers and send that voltage to the ADC of the MicroMod. The MicroMod takes these signals from the user inputs and translates them into XInput standard USB communication, which then gets sent to the computer to process. All these inputs are laid out on one flat surface so that you do not need to hold the FXC while playing with it.

Motivation
The Fingerless Xbox Controller was made for users with a wide range of hands. From personal experience, family game night can be difficult when one player cannot reach all the buttons. The Fingerless Xbox Controller was designed to make gaming more fun and accessible for everyone.