Chase Williams

Senior Design 2

Journal for 2/27/2022

Touch Detection on All Keys

What is the problem: The code I created to detect touches on all 61 keys was not functioning as I intended it to when I met with the hardware group members to test it. When someone would touch one of the touch nodes the code would indicate that node was touched along with another node that was not touched.

Why this is a problem: Without being able to correctly indicate which node was touched, we would only be able to add touch detection to 48 keys not 61.

How I solved it: With how we configured the addresses for the six capacitive touch sensors, we are only using three addresses: 0x5A, 0x5B, and 0x5C. In addition, we have three sensors connected to each of the two multiplexer ports. When pin 0 on 0x5A from the first multiplexer port was touched, I noticed that code indicated this pin was touched as well as pin 0 on 0x5A from the second multiplexer port. I also noticed that I called the choose port function in the part of the code that loops through each sensor port to see which pin was touched and which one is released. However, there were places where I reassigned values to the variables tied to each sensor but did not call the choose port function. After I called the function in these places, this issue was resolved. I needed to call the choose port function every time I referenced variables tied to a sensor.

Determine how a change in resistivity impacts the Change in ADC counts

What is the problem: Although we made our assumptions, the group was not sure how a change in resistance attached to a touch node would impact the outputted ADC counts.

Why this is a problem: We needed to decide what type of conductive spray to buy and noticed there were different resistivities in each option. We wanted to buy the spray that would either increase the difference between the baseline and filter ADC count values or just not lead to a decrease in the difference of ADC counts.

How I solved it: I added resistance to the capacitive touch sensing nodes and collected the ADC counts data to see if this had an impact on the ADC counts values. Here the results I got and a summary. The conclusion I reached is we need to buy a spray with less resistance and resistivity because this leads to a greater increase in the different in ADC counts between the baseline

and filter values when a node is touched, which gives us a better chance of detecting touches through a nonconductive material.

Added Resistance	Average Baseline ADC counts (No Touch)	Average Filter ADC counts (No Touch)	Difference (No Touch)	Average Baseline ADC counts (Touch)	Average Filter ADC counts (Touch)	Difference (Touch)	Average Baseline ADC counts (Touch-No Touch)	Average Filter ADC counts (Touch-No Touch)
50 ohms	224	226	2	222	59	163	2	167
25 ohms	236	236	0	235	52	183	1	184
0 ohms	278	282	4	276	60	216	2	222

Summary

- As the resistance decreases:
    - When there's no touch:
        - The baseline and filter values increase and are about the same.
        - The difference between the baseline and filter values is about 0.
    - When there's a touch:
        - The baseline ADC counts value increases.
        - The filter ADC counts value remains fairly constant or there's no correlation.
        - The difference between the baseline and filter values increases.

Test changing touch detection sensitivity code

What is the problem: I have not confirmed whether the code that allows someone to adjust the touch detection sensitivity works.

Why this is a problem: If we cannot adjust the sensitivity of the touch detection, we may not be able to detect touches through a nonconductive material.

How I solved it: I learned that the default setting for the touch detection threshold is 12 ADC counts and the release detection threshold is 6 ADC counts. The threshold is the difference in ADC counts between the baseline and filter values. After changing the thresholds to 250 and 125 for touch and release detection respectively, I observed that the sensors were not able to detect touches. In addition, after changing the thresholds to 2 and 1 for touch and release detection respectively, I observed that the sensors able to detect touches even through nonconductive material. If I only touched the wire insulation, a touch was detected. This reduction in thresholds did cause the code to indicate a touch or release occurred when it did not happen in reality.

Journal for 2/13/2022

Topic: Touch and release detection

What is the problem:

The code we currently have for the microcontroller (Arduino Nano) will not be able to detect touches and releases that occur on the outside surface of a keyboard key from metal connected to the capacitive sensing node and the inside surface of the keyboard key.

Why this is a problem:

If we are unable to detect touches on the outside surface of a key with metal on the inside surface of the key, we will need to change the design, which could lead to delays in our product development. A couple alternative options we would then need to consider are buying metal keycaps or applying a conductive coating to the outside surface of each key.

How I am planning to solve it:

I will work on calibrating the touch detection software through a trial-and-error method, so we can figure out the correct thresholds that determine when a user has touched the key and when their finger is lifted off the key. There is a "setTresholds" function that allows you to enter the ADC counts touch and release thresholds. I will start with something small like 4 for both the touch and release thresholds and view the accuracy when the touch detection hardware is under a key. I will change these thresholds until I get to 100% accuracy for both touch and release detection.

Topic: How to route wires to from the sensors to the inside surface of each key cap

What is the problem:

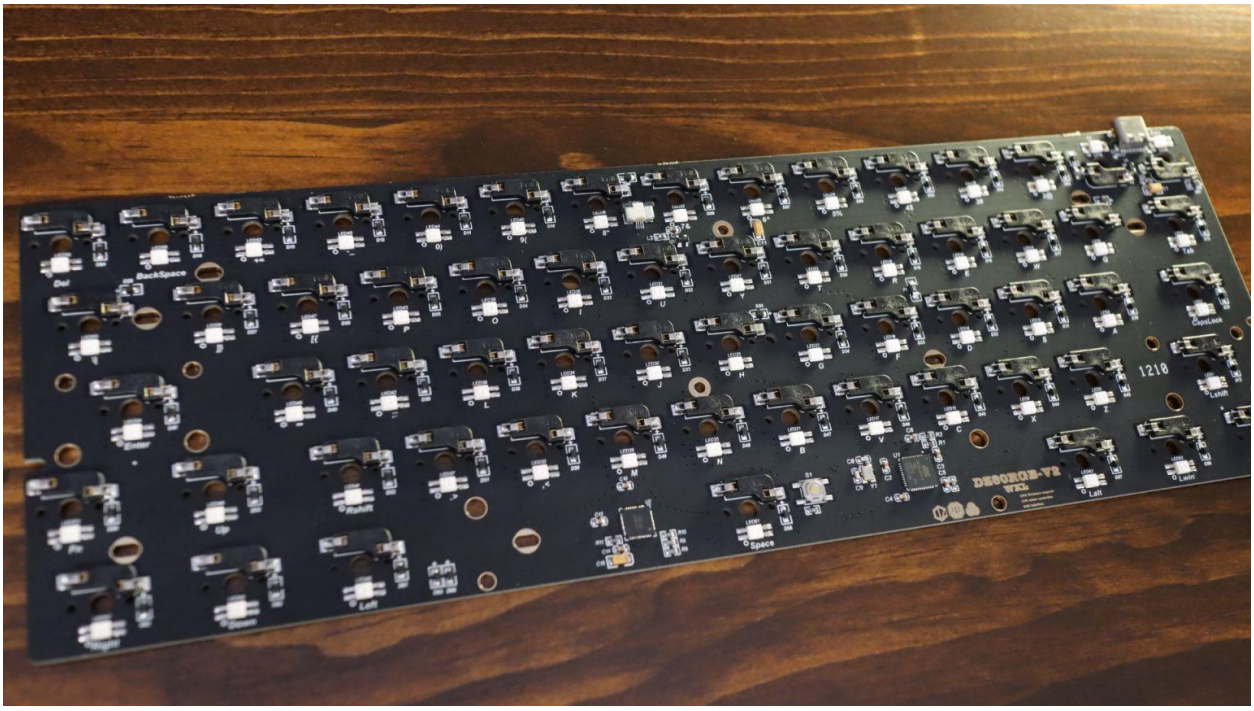
We have not figured out a method to create a metal connection from the touch sensors through the keyboard PCB and mechanical switches for the keys to the inside surface of each key cap.

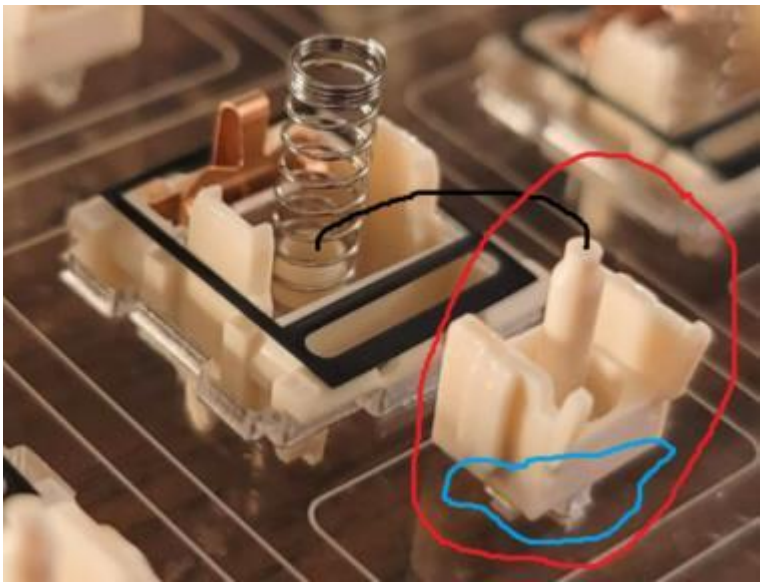
Why this is a problem:

If we are unable to create a metal connection from the touch sensors through the keyboard PCB and mechanical switches for the keys to the inside surface of each key cap, we will be unable to have all the hardware is not visible to the user. Wires connecting to all 61 keys would be showing, which would not be aesthetically pleasing. In addition, this could lead to our device being fragile and not having great durability because of the exposed wires.

How I am planning to solve it:

The solution I presented to the group was to find a keyboard PCB with holes where the keyboard keys should be placed like in the example below. We could then coat the piece circled in red (in the second image) with metal, possibly using a conductive spray. The black line is the wire that goes through the hole to the bottom of the piece coated with metal. A hole may need to be drilled into the mechanical switch to feed the wire through the hole as I have illustrated. Finally, we can add metal connection from the top of the piece (in blue) to the inside surface of the key. This design would allow us to have a metal connection from the hardware to the inside surface of the keycap and would let us hide the wires from the user.





Topic: Integrating the microcontroller, multiplexer, and touch sensors

What is the problem:

While we do know how to connect the touch sensors to the microcontroller, we have not figured out how to integrate the microcontroller, multiplexer, and touch sensors.

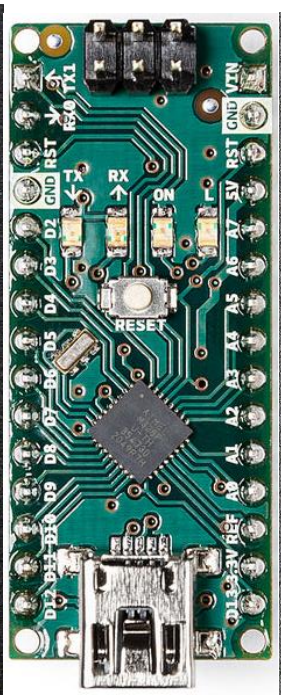
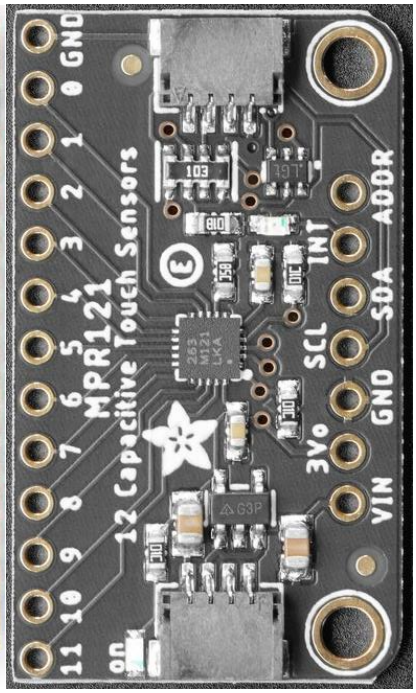
Why this is a problem:

If we are not able to integrate the multiplexer in our touch detection circuit, we could be at risk for not being able to provide touch detection capabilities to all keyboard keys on a 60% keyboard. At most we could detect touches on 48 keys because there is only one I2C bus on the microcontroller we have. Consequently, we would fail to meet our requirement of touch detection on 61 keys.

How I am planning to solve it:

After doing some research, I have figure out how to integrate the microcontroller, multiplexer, and touch sensors. Going from left to right, I have added picture of the multiplexer, microcontroller, and touch sensor, respectively. This allows the pin names to be seen on all three boards. The connections between the devices are outlined below.

- Vin
    - If you connect all sensors to 5V, Vin on multiplexer goes to 5V on Nano
    - If you connect all sensors to 3.3V, Vin on multiplexer goes to 3.3V on Nano
- GND
    - GND on multiplexer goes to GND on Nano all the capacitive sensors are connected to
- SDA
    - SDA on multiplexer connects to A4 on Nano
    - I2C on Nano: A4 (SDA) and A5 (SCL)
- SCL
    - SCL on multiplexer connects to A5 on Nano
    - I2C on Nano: A4 (SDA) and A5 (SCL)
- SDN
    - Connect SDN on first four sensors to SD0 on the multiplexer
    - Connect SDN on the rest of the sensors to SD1 on the multiplexer
- SCN
    - Connect SCN on first four sensors to SC0 on the multiplexer
    - Connect SCN on the rest of the sensors to SC1 on the multiplexer


VIN
GND
SDA
SCL
RST
A0
A1
A2
SD0
SC0
SD1
SC1
SC7
SD7
SC6
SD6
SC5
SD5
SC4
SD4
SC3
SD3
SC2
SD2
TCA9548A
I2C
Expander

RESET
TX
RX
ON

MPR121
12 Capacitive Touch Sensors
GND
SCL
SDA
ADDR
INT
3Vo
VIN