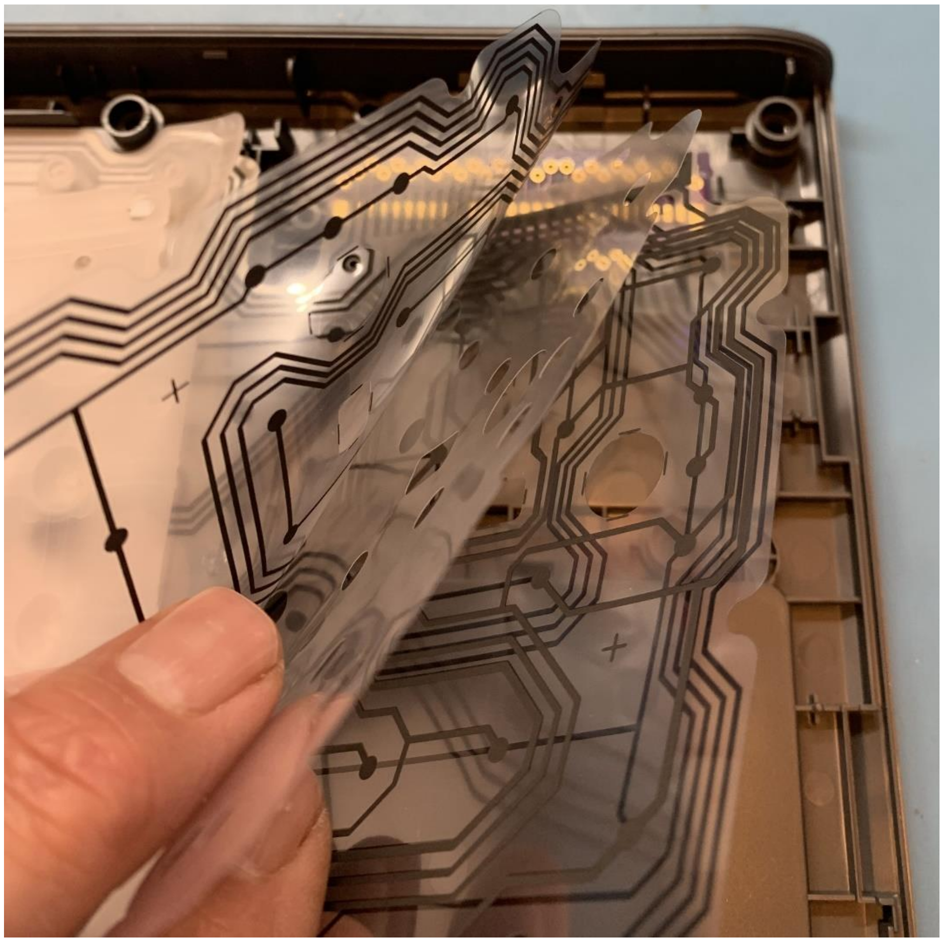
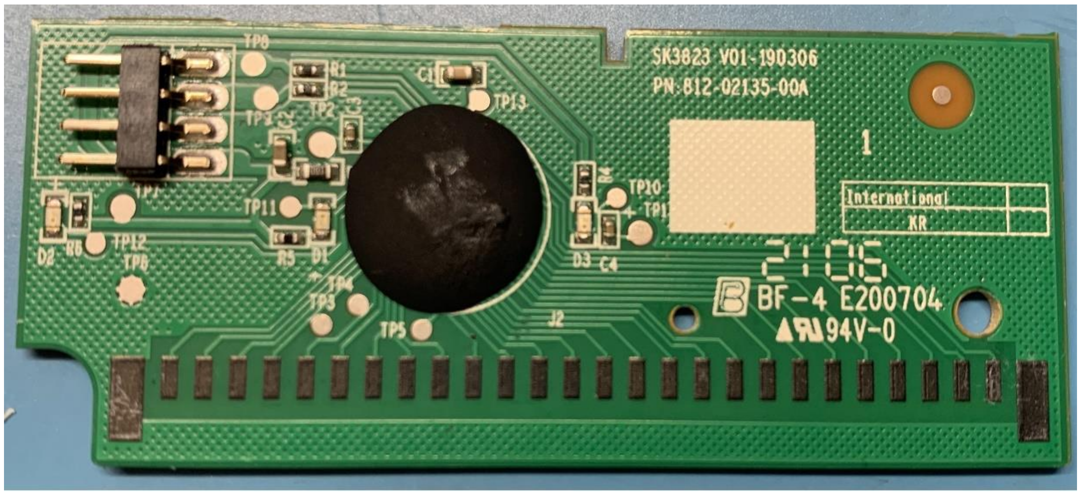
Logitech Keyboard K120

This project replaced the original controller inside the Logitech K120 keyboard with a Teensy 4.1. The Teensy would not fit in the space were the original controller was housed so the row/column wires were brought out of the case so the Teensy could be wired externally.

The row and column signals from the key switches are made with traces on plastic sheets as shown below.

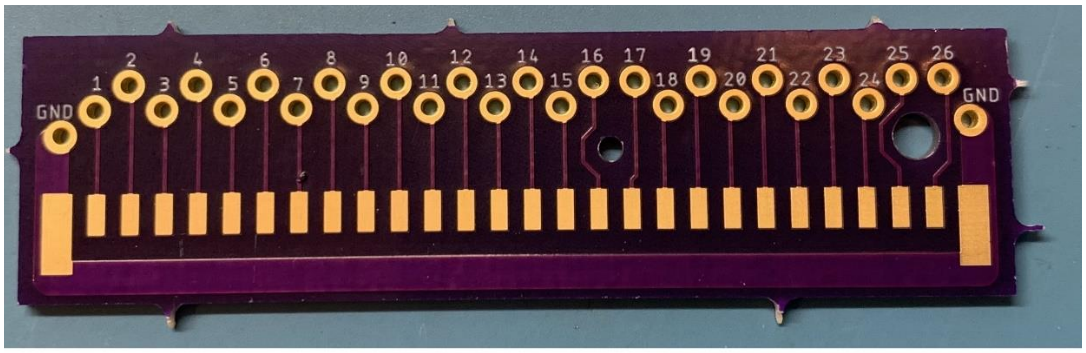


The key traces come to the upper right corner of the keyboard where they previously connected to a small circuit board with a blob top controller chip. I took measurements with a micrometer so I could match the mounting holes and trace locations with a new PCB.



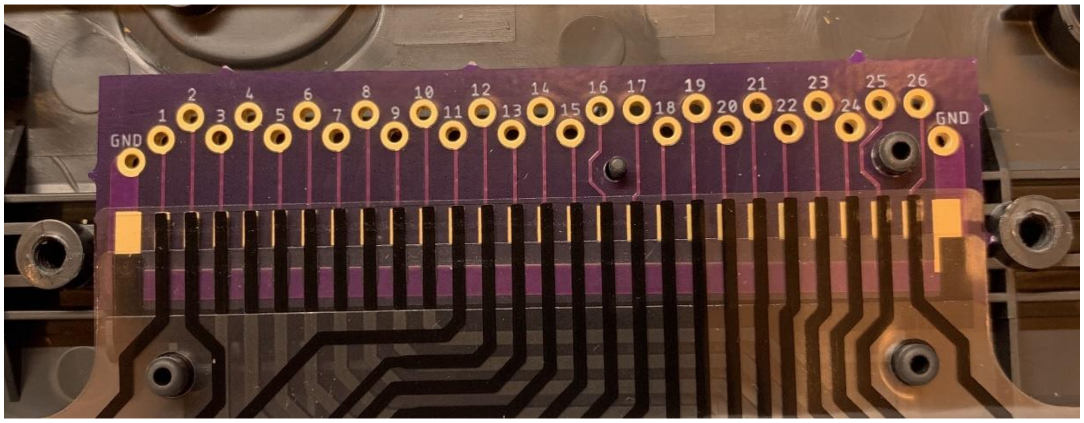
Original keyboard controller "blob top" IC

The OSH Park PCB was made from Eagle file "Logitech_K120.brd". It only has connections for the 26 signal traces and ground.

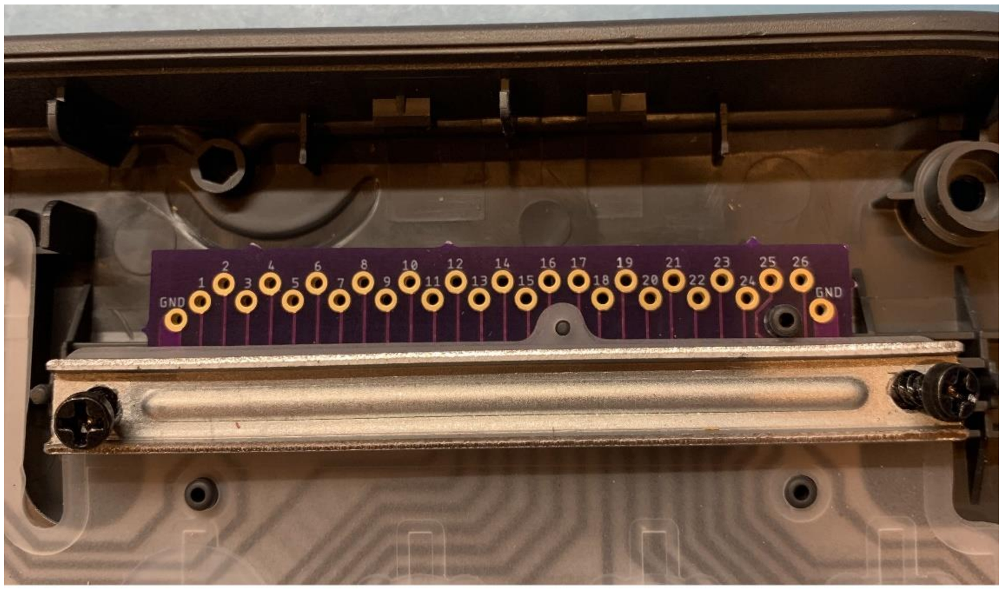


New PCB

The traces on the plastic sheets extend over the pads on the PCB. The picture below shows I could adjust the PCB traces slightly to the right but its good enough to make a connection.

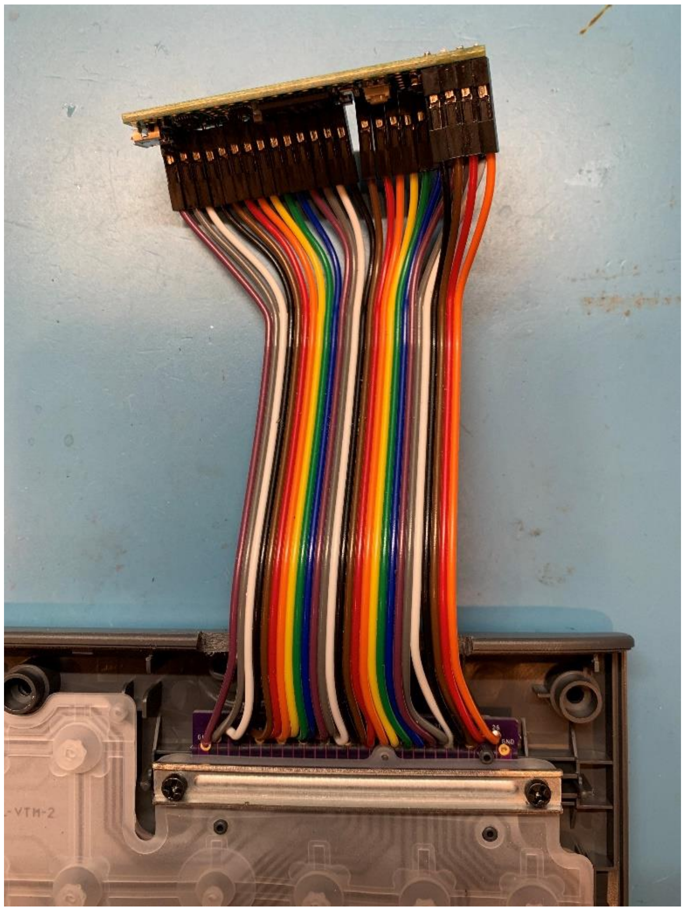


The traces are connected together by screwing down a metal bar to mash them together.

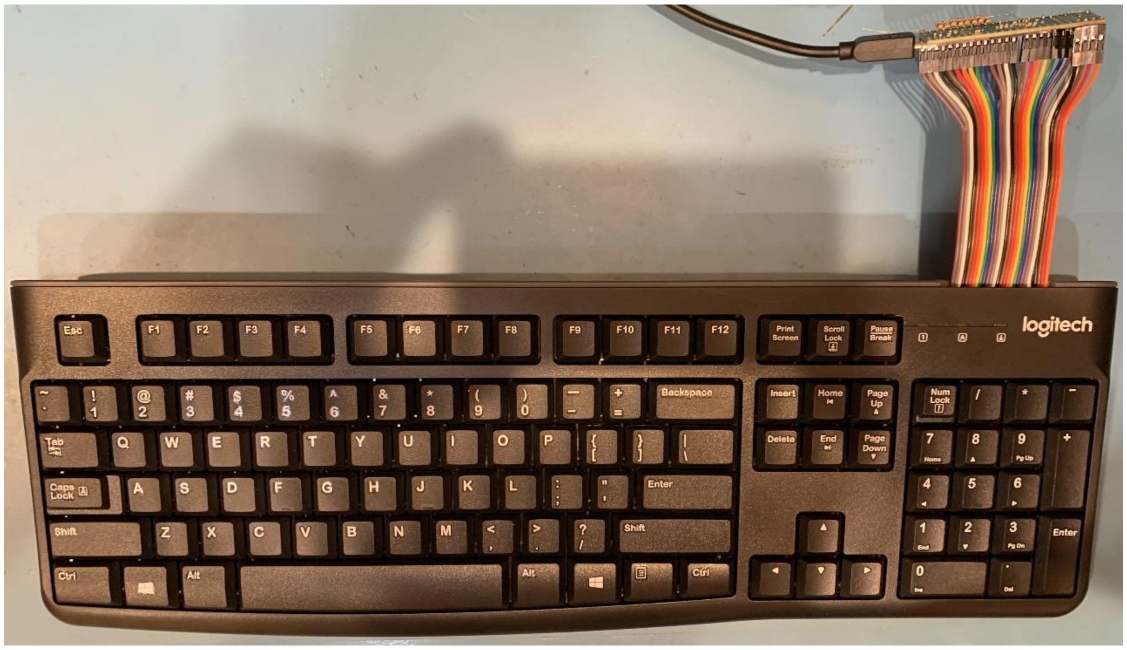


Signal traces pressed against pads on PCB using metal bar

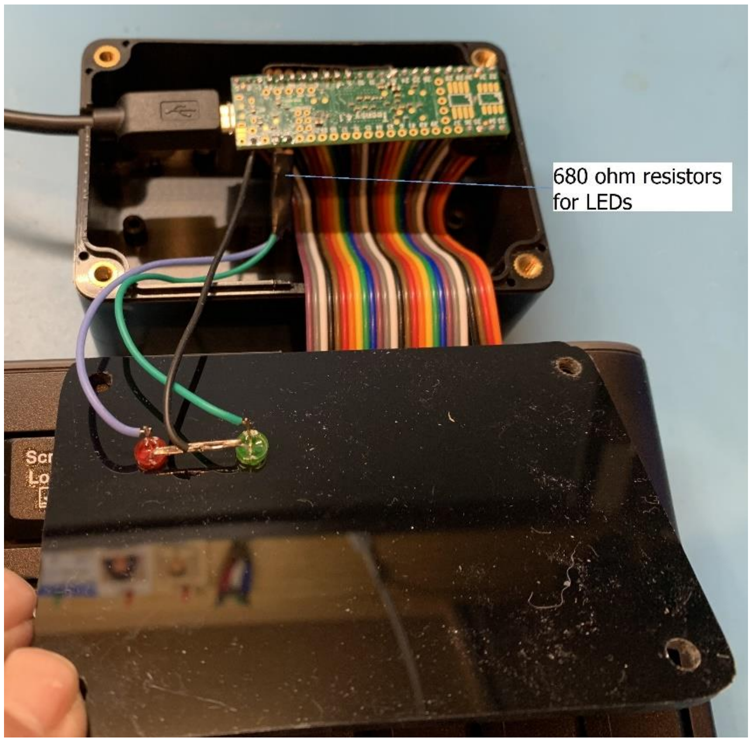
The 26 signals plus ground were soldered to the PCB and routed to a Teensy 4.1. A Dremel tool was used to trim the plastic on the keyboard case so the wires could escape.



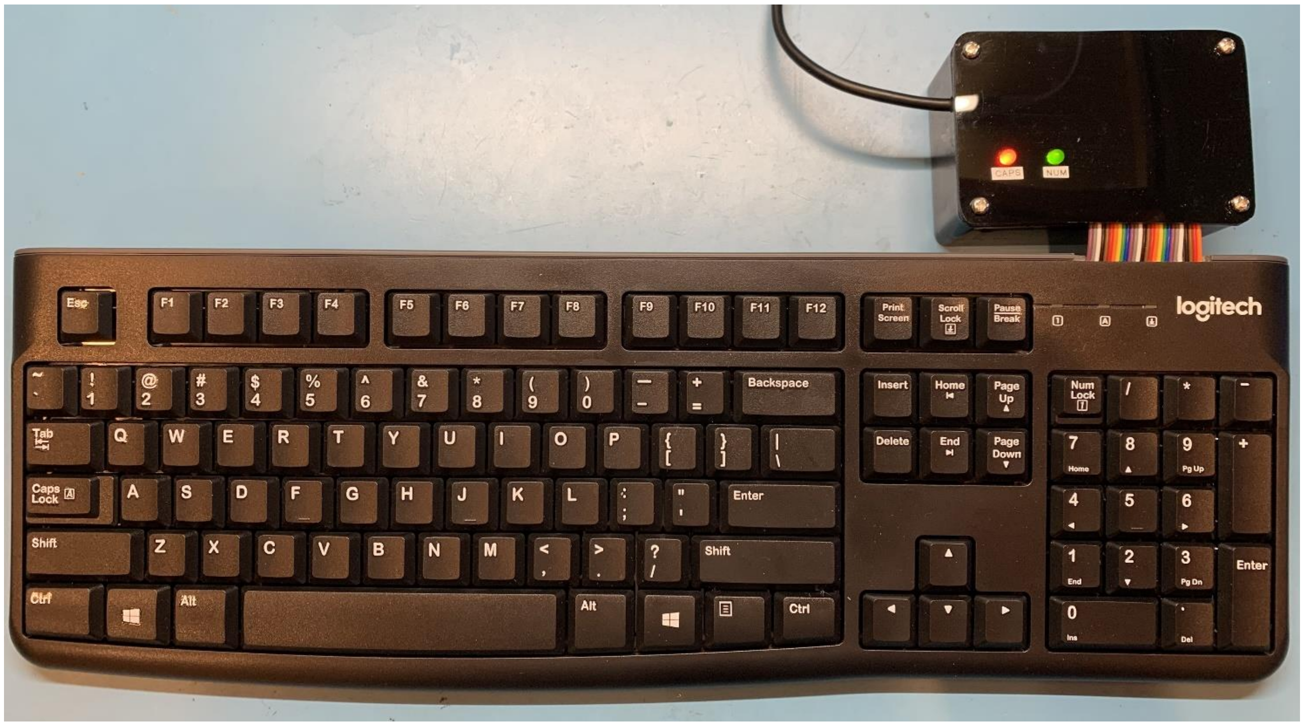
This shows the keyboard put back together with wires to the Teensy.



I used a plastic project box to house the Teensy. The lid is cut from a sheet of acrylic and has holes for CAPS and NUM Lock LEDs. The USB cable enters the box thru a slit on the side.



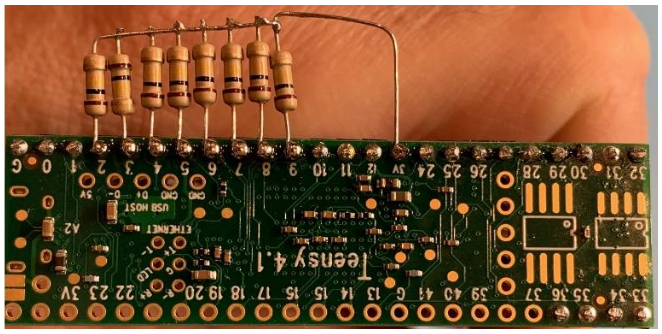
This shows the Teensy project box with both LEDs lit prior to adding the ErgoTrac pointing device.



The table below gives the wiring from the PCB to the Teensy 4.1.

PCB	Teensy I/O
1	0
2	1
3	2
4	3
5	4
6	5
7	6
8	7
9	8
10	9
11	10
12	11
13	12
14	24
15	25
16	26
17	27
18	28
19	29
20	30
21	31
22	32
23	36
24	35
25	34
26	33

I tried to use my matrix decoder code but it would only give the pin connections for the keys on the right side of the keyboard. This was caused by the high resistance traces on the plastic sheets. For example, the “X” key traces measured over 32K ohms at the Teensy yet the keypad “-“ key measured 2K ohms. Voltage measurements showed the “X” key would only go down to 1.6 volts when pressed because the Teensy code uses the internal 33K ohm pullup resistors. This voltage was not seen as a logic low by the Teensy. I determined the 8 column pins by visually following the traces on the plastic sheets going to the modifier keys. Then I soldered 100K ohm pull up resistors to the 8 column inputs (see below). I also modified my standard USB keyboard controller code to not use the internal 33K ohm Teensy pull ups.



The matrix given below was determined by temporarily loading a row in the normal array with KEY_1 thru Key_8. Then I pressed every key looking for a number to be output. The number was then replaced with the actual key name. The PCB pad number is given first and the Teensy I/O number second. Note the keypad keys start with KP. The Teensy code for the keyboard is named “Logitech_K120_4p1.ino”

PCB-IO	**3-2**	**4-3**	**5-4**	**6-5**	**7-6**	**8-7**	**9-8**	**10-9**
1-0			Cntr-L		Cntr-R			
2-1	Esc	Tilde	Tab	Q	A	F	Z	CAPS
11-10	F8	F9	8	9	0	O	L	/
12-11	PRNTSCR	SCRL	PAUSE	PAGEUP	NUM	HOME	PAGEDN	INS
13-12	F10	F11	F12	MINUS	EQUAL	I	LEFT	
14-24	F6	F7	U	H	7	J	N	PERIOD
15-25	F4	4	R	R	5	F	V	M
16-26	F5	T	6	Y	G	B	COMMA	4
17-27				Shift-L				Shift-R
18-28	SLASH	KP-8	END	KP-*	KP-7	KP-9	KP-5	KP-6
19-29		GUI-L						
20-30		BCKSPC	P	[	;	QUOTE	K	DOWN
21-31	]	\	DEL	ENTER	UP	RIGHT	KP-0	SPACE
22-32	KP-4	KP-1	KP-2	KP-3	KPMINUS	KP-PLUS		KP-ENTR
23-36	F2	F3	3	E	D	C		
24-35	F1	1	2	W	9	X	S	KP-.
25-34	Alt-L					Alt-R		
26-33							Fn (GUI-R)	

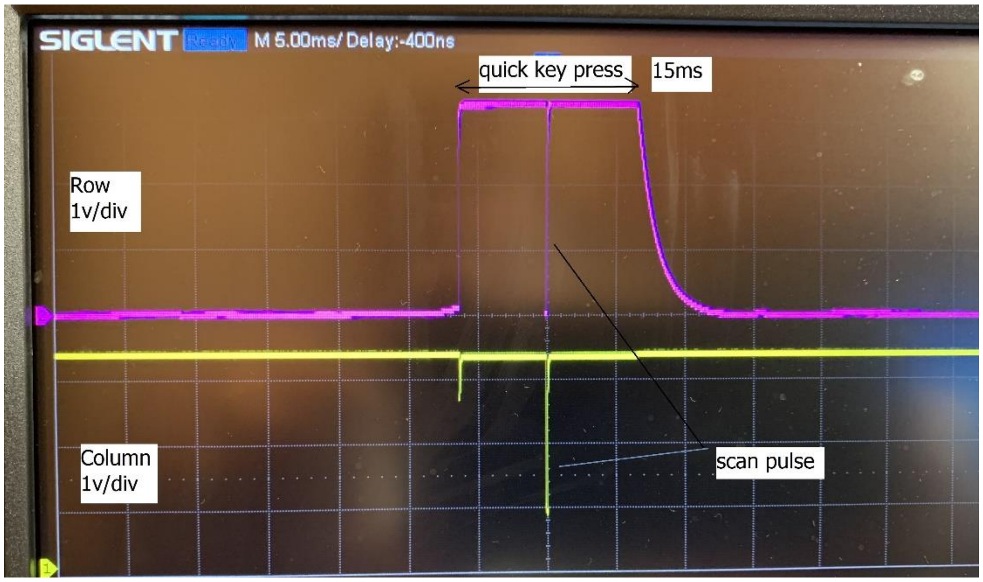
This keyboard has no Fn modifier key but it does have two Windows GUI keys so I replaced the GUI-R key with Fn. This allowed me to add media controls as follows:

Fn-F1 = Mute

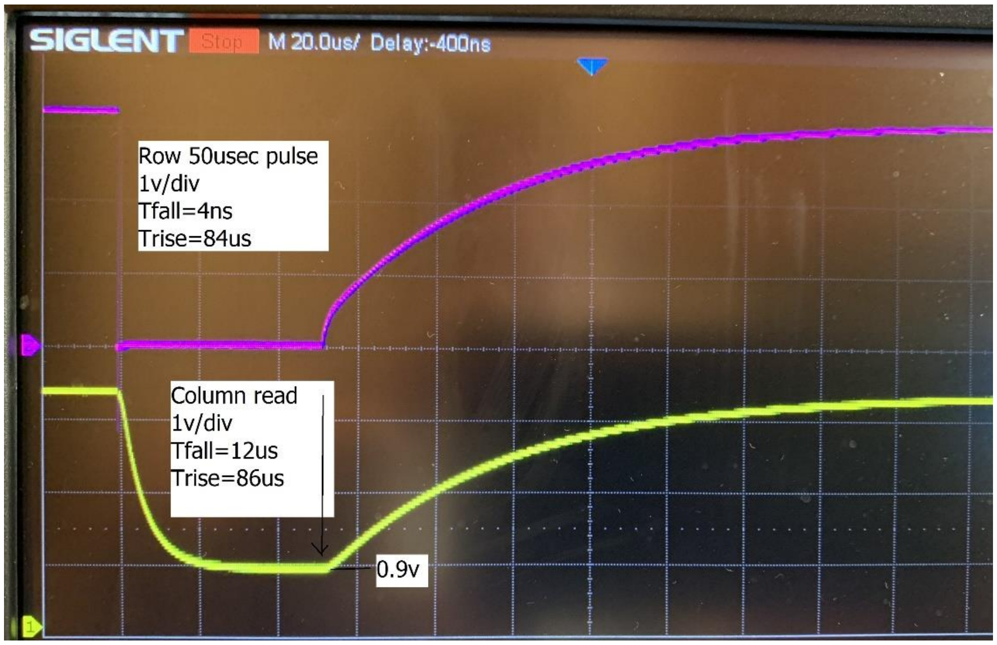
Fn-F2 = Volume Down

Fn-F3 = Volume Up

The picture below shows the Teensy row and column when the “X” key is pressed very fast.



Multiple attempts were tried and the fastest keypress was 15msec. The row signal starts out floating at 0 volts and the column signal starts out pulled up to 3.3 volts with a 100K Ohm resistor. When the X key is first pressed, the floating row signal is connected to the pulled up 3.3 volt column, causing the row to also go to 3.3 volts. Later the row is driven low by the Teensy so the column input can be read. The picture below shows the scan pulse with more detail. The column signal at the Teensy has a fall time of 12usec because the “X” key traces and switch measure 32K ohms. The 0.9 volt logic low signal is read by the Teensy after the row has been low for 50usec, just before the row is floated. Once allowed to float, the row and column (which are still connected together by the switch) are slowly pulled up to 3.3 volts by the 100K ohm external resistor.



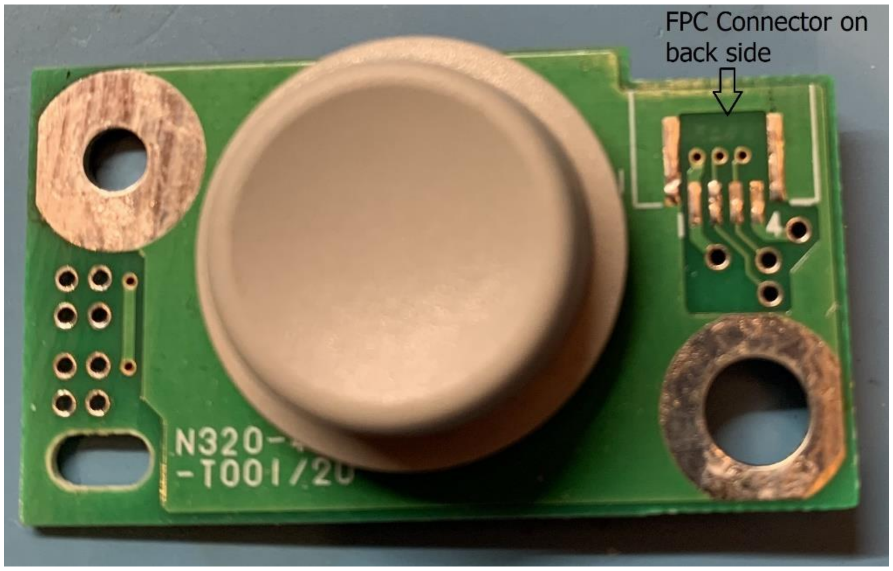
An ErgoTrac pointing device, marked N320-4828-T001/20 is shown below. This one came from EBay, and is sold as FID-828-100/20. These pointers are used in certain Fujitsu Laptops and may have the FPC connector on the top instead of the bottom. I used 4 discreet wires instead of an FPC cable to hook up to the Teensy 4.1. I've also used a Teensy 3.2 with this device as documented here. The pinout for the FPC connector and the wire routing to the Teensy is as follows:

FPC Pin 1 = 3.3 volt reference pulse wired to Teensy I/O 39

FPC Pin 2 = X direction wired to Teensy ADC 17

FPC Pin 3 = Y direction wired to Teensy ADC 16

FPC Pin 4 = Ground wired to Teensy ground

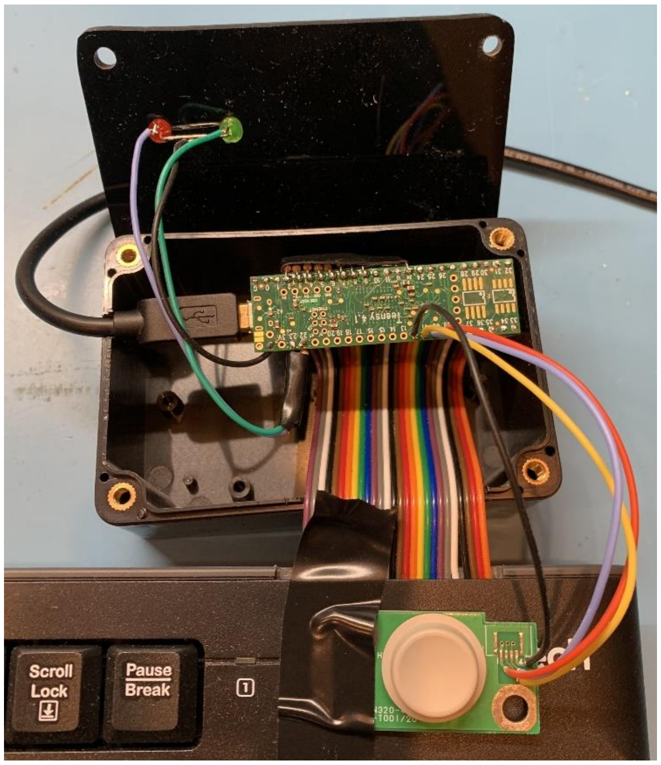


After completing the keyboard scan, the Teensy 4.1 sends I/O pin 39 high to 3.05 volts which translates to a voltage on pins 2 and 3 that varies based on movement of the pointing head. These voltages are converted by the ADC in the Teensy and compared to the "at rest" values taken at power up. The ADC takes 8 readings and averages to filter noise. Once finished, the Teensy sends I/O pin 39 low because this signal needs to be a pulse, not a constant level. The following table gives the X and Y pulse voltages at rest and pushed hard in each direction:

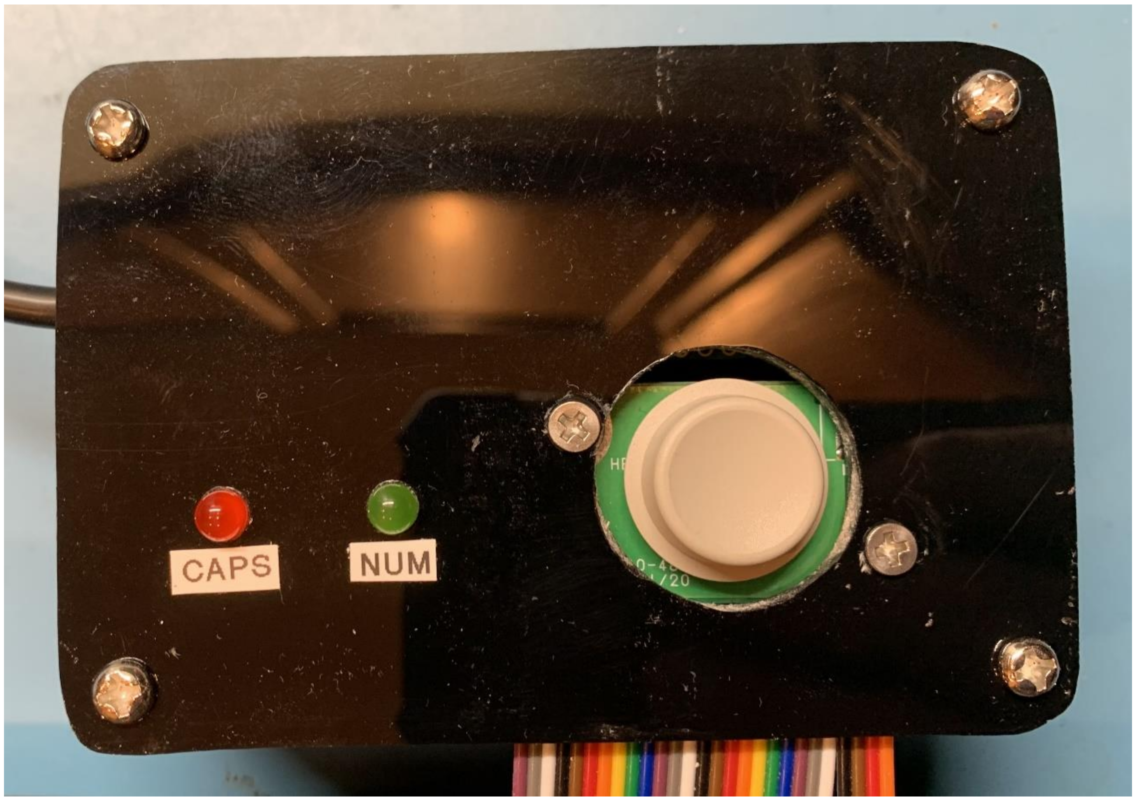
	Pin 2 X Direction	Pin 3 Y Direction
At Rest	1.6 V	1.4 V
Hard Left	1.1 V	
Hard Right	2.2 V	
Hard Down		0.75 V
Hard Up		2.05 V

The Teensy code saves the "at rest" X and Y values at startup, then it scans the keyboard and ErgoTrac about every 30mec. X or Y readings from the ErgoTrac that are beyond the "at rest" values (with an added noise value) are converted to relative mouse movement values and sent over USB.

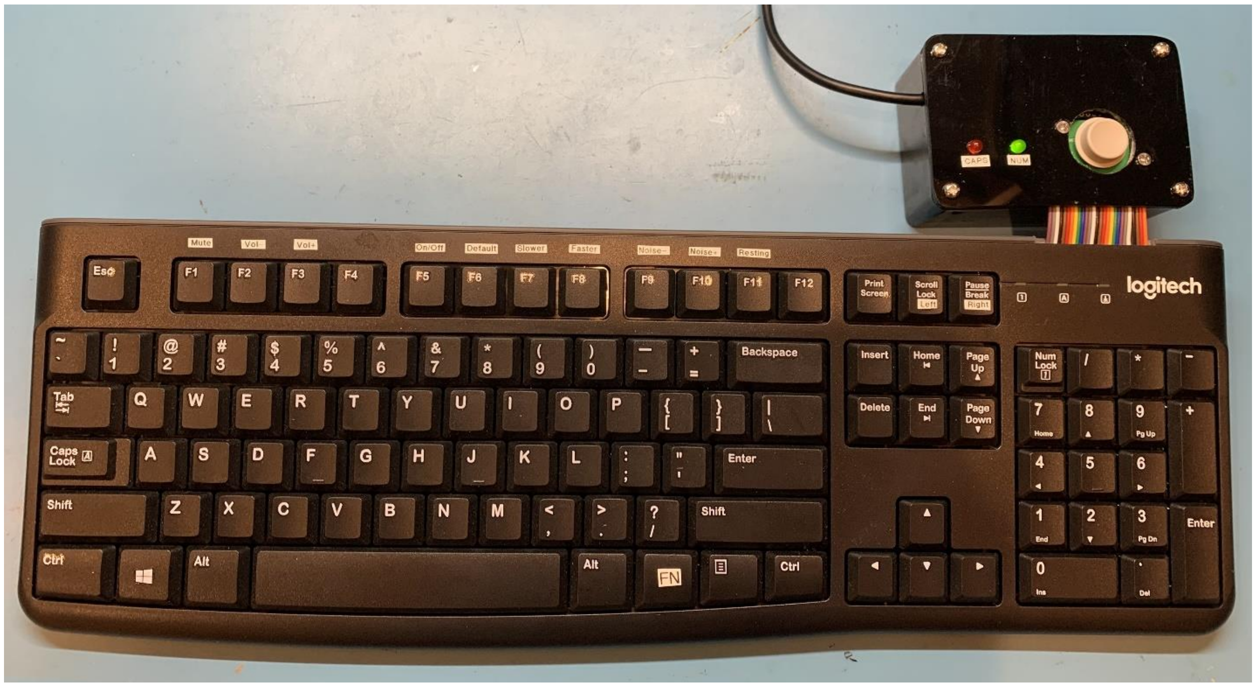
This picture shows initial testing of the ErgoTrac with the Teensy 4.1.



This shows the ErgoTrac inside the box with a hole in the lid.



This is the finished keyboard with ErgoTrac. I've added labels for special keys. I never use Scroll Lock or Pause so these keys have been repurposed as left and right mouse buttons to go along with using the ErgoTrac.



The ErgoTrac works fine with the default values but if I want to try adjusting the settings, the following Fn-function keys can be used. The right Windows key acts as the Fn key.

Fn-F5 = Toggle ErgoTrac On/Off. Wakes up turned On. Left/right mouse buttons always function

Fn-F6 = Change Speed and Noise Zone back to the default values used at startup

Fn-F7 = Decrease cursor speed (takes several presses to notice a difference)

Fn-F8 = Increase cursor speed (takes several presses to notice a difference)

Fn-F9 = Decrease Noise Zone (too small and the cursor will move on its own due to noise)

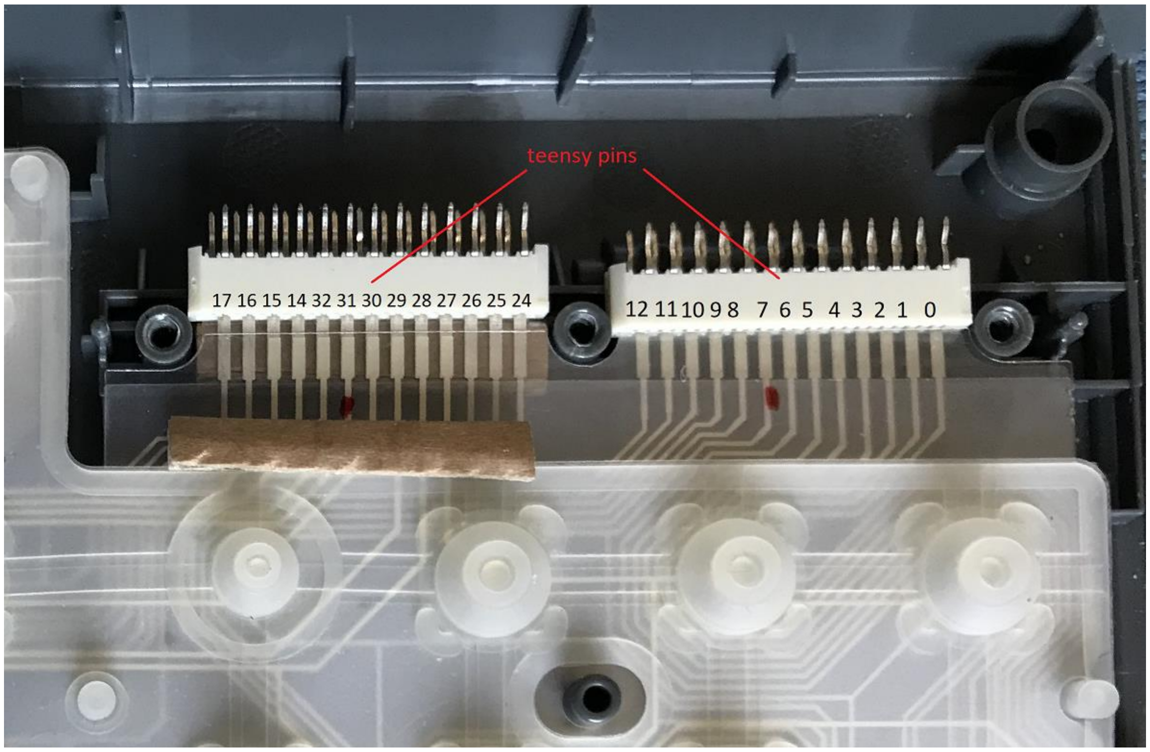
Fn-F10 = Increase Noise Zone (too big and it will take excessive pressure to get any movement)

Fn-F11 = Recapture the ErgoTrac "at rest" X and Y values. Use this if temperature changes over time cause the cursor to start drifting.

The USB Composite keyboard and ErgoTrac code is called "Logitech_K120_4p1_ergotrac.ino" and it can be downloaded from my repo.

The older version of Logitech K120 keyboard shown below was converted to USB by Radu.

This version keyboard uses a different material for the circuit traces that are not as resistive, allowing the normal matrix decoder code to give a complete pin connection list. Top and bottom contact FPC connectors were used with wires soldered to every other pin.



The pins were wired to the Teensy 4.1 I/O's as listed above. The key matrix is different so it uses different code, named "Logitech_K120_previous_version.ino". Quick key presses were sometimes missed so the overall loop rate was reduced. Unfortunately this caused a double key press to sometimes be seen. The code includes an attempt at a switch debounce/noise filter but it has not fixed the problem. Further debugging is planned.