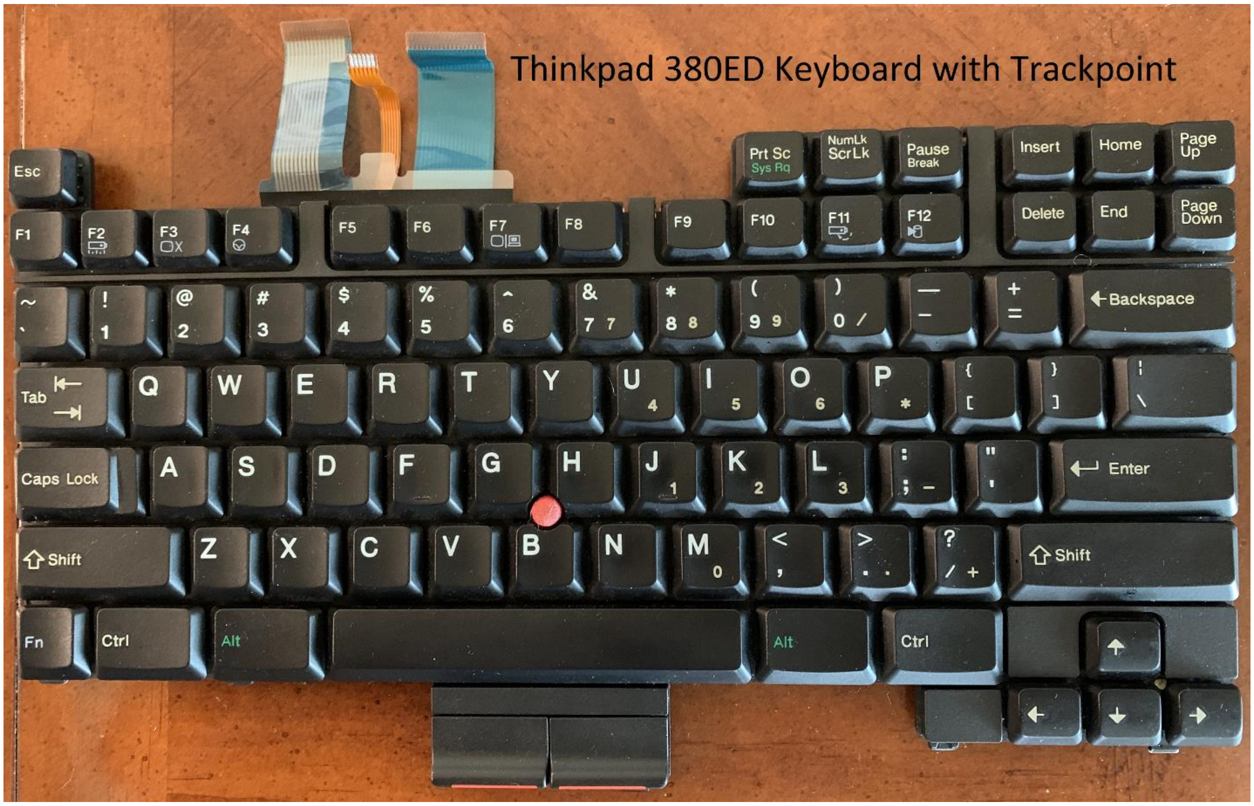
IBM Thinkpad 380ED Keyboard Project

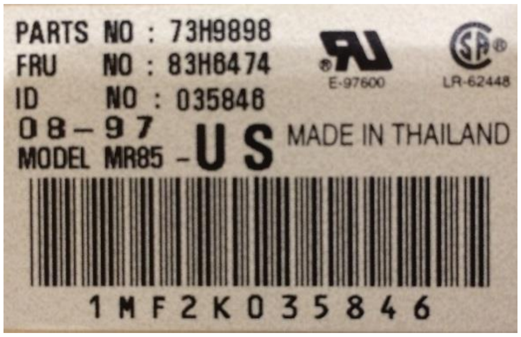
Introduction: Older Thinkpad keyboards have full size keys that feel much better than the “chicklet” keys on modern laptops. A nice feature of these keyboards is the trackpoint eraser head that takes a lot less space than a touchpad. I have previously converted a Thinkpad T61 keyboard and trackpoint to USB with a Teensy microcontroller as documented in my Instructable. Thinkpad keyboards from models T6X, T4X, T2X and their variants have a built in circuit that converts the trackpoint resistive strain gauges to PS/2. Older Thinkpads, like the 380ED, rely on circuitry external to the keyboard to do the conversion.

Project Overview: This project will develop the circuitry to amplify the voltage changes from the trackpoint strain gauges on a Thinkpad 380ED keyboard (shown below). A Teensy microcontroller will detect the voltage change using it’s A to D converter. The keyboard key matrix will also be scanned by the Teensy to create a composite USB keyboard and mouse device.



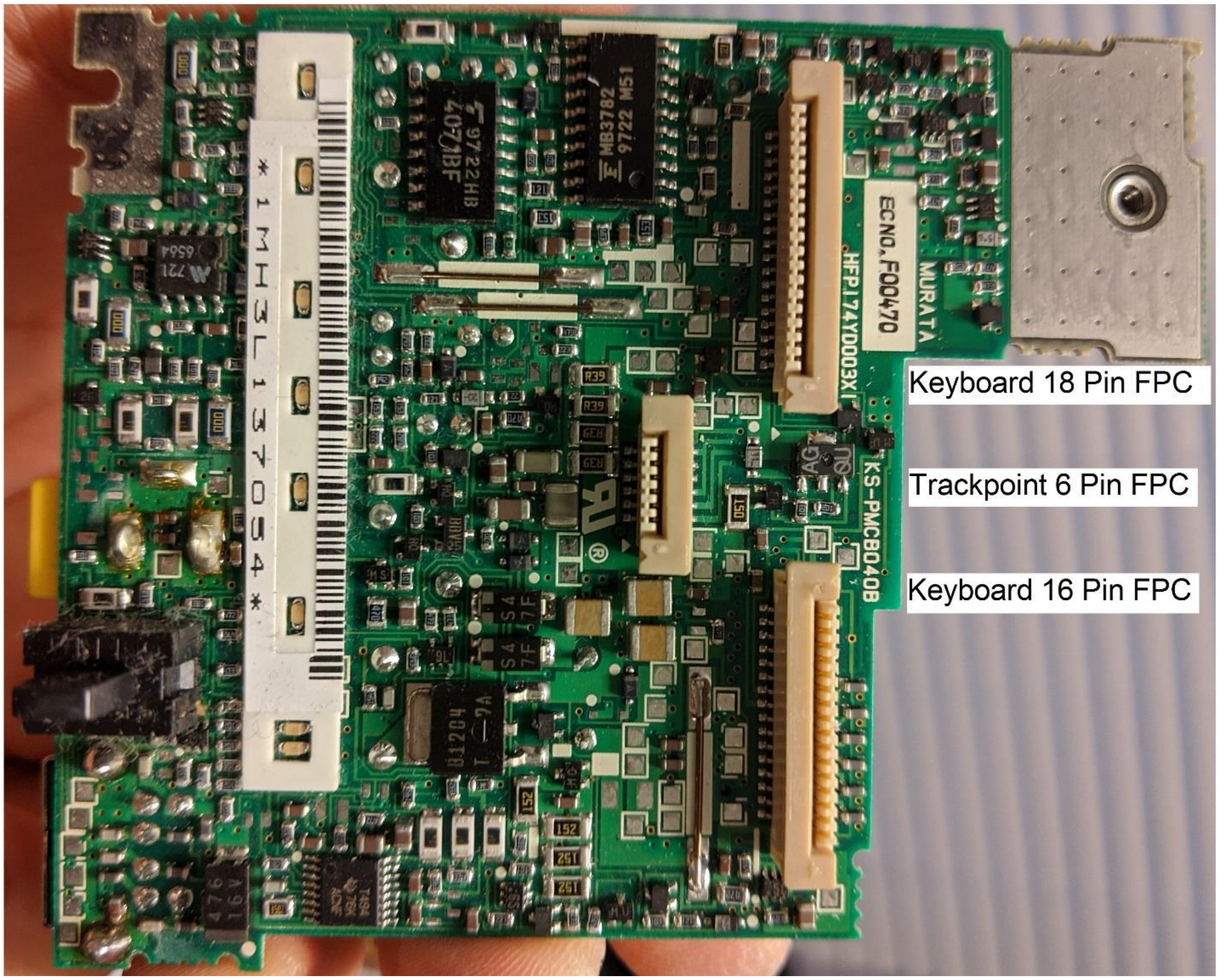
Thinkpad 380ED Keyboard with Trackpoint

Keyboard Part Number.

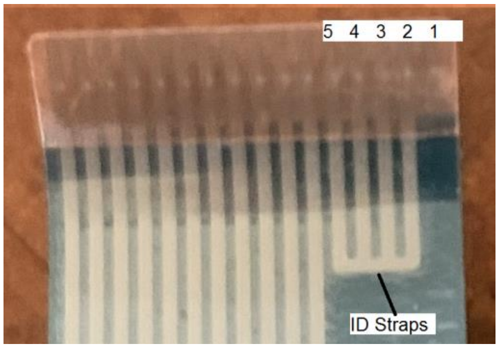


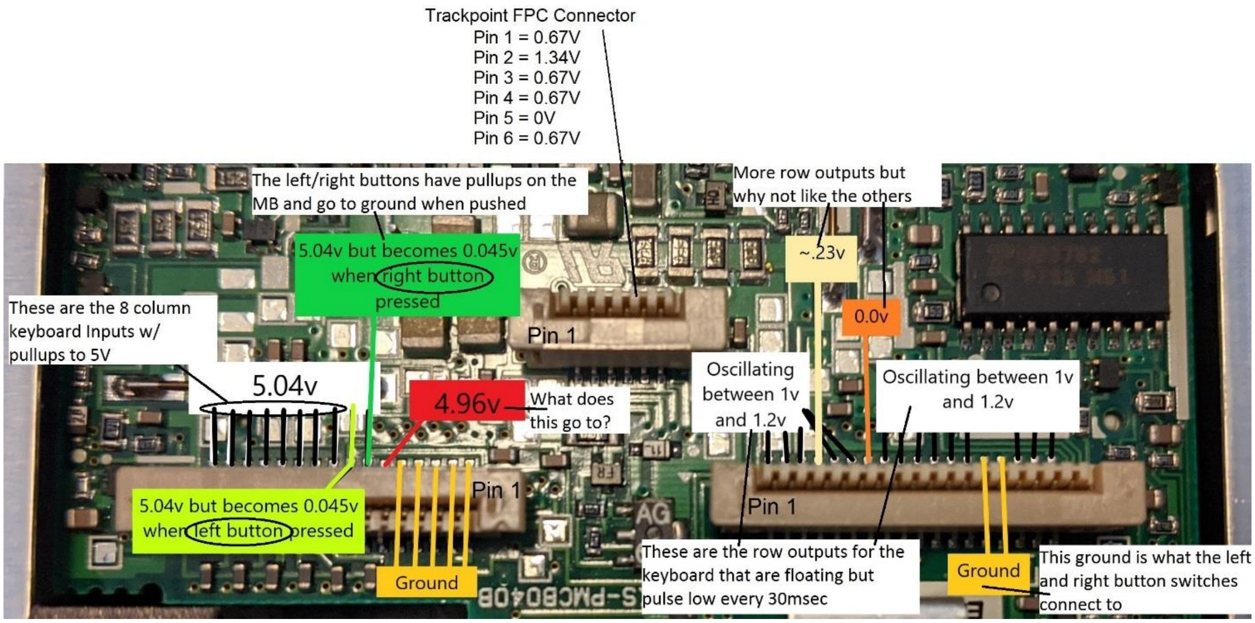
PARTS NO : 73H9898
FRU NO : 83H6474
ID NO : 035846
08–97
MODEL MR85 - U S MADE IN THAILAND
E-97600
LR-62448
1MF2K035846

Initial Testing: The 3 FPC cables from the 380ED keyboard connect to the power supply/keyboard daughterboard (shown below). I could not find a schematic for this laptop so resistance and voltage measurements were used to figure out the connections.



The 3 FPC cables were installed and the daughterboard plugged into the motherboard in order to measure the voltages on the connector pins. Note that the first 5 FPC traces on the 16 pin cable don't go to the keyboard. Pin 1 is open and pins 2 thru 5 are tied together as shown below.

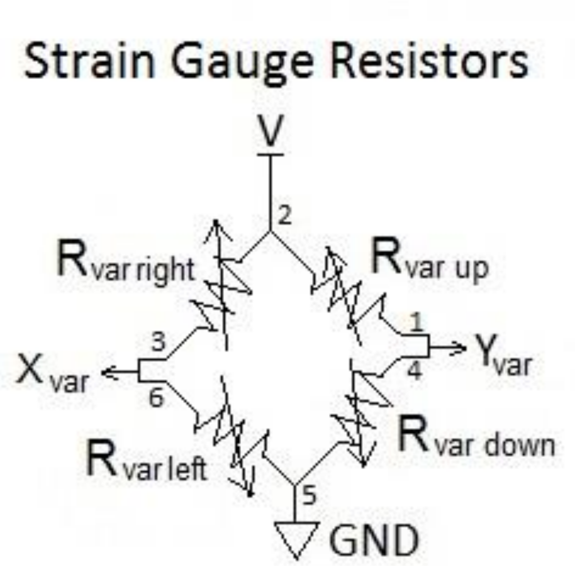



Trackpoint FPC Connector
Pin 1 = 0.67V
Pin 2 = 1.34V
Pin 3 = 0.67V
Pin 4 = 0.67V
Pin 5 = 0V
Pin 6 = 0.67V
The left/right buttons have pullups on the MB and go to ground when pushed
More row outputs but why not like the others
~.23v
0.0v
5.04v but becomes 0.045v when right button pressed
These are the 8 column keyboard Inputs w/ pullups to 5V
Pin 1
5.04v
4.96v
What does this go to?
Oscillating between 1v and 1.2v
Oscillating between 1v and 1.2v
Pin 1
5.04v but becomes 0.045v when left button pressed
Pin 1
Ground
These are the row outputs for the keyboard that are floating but pulse low every 30msec
Ground
This ground is what the left and right button switches connect to

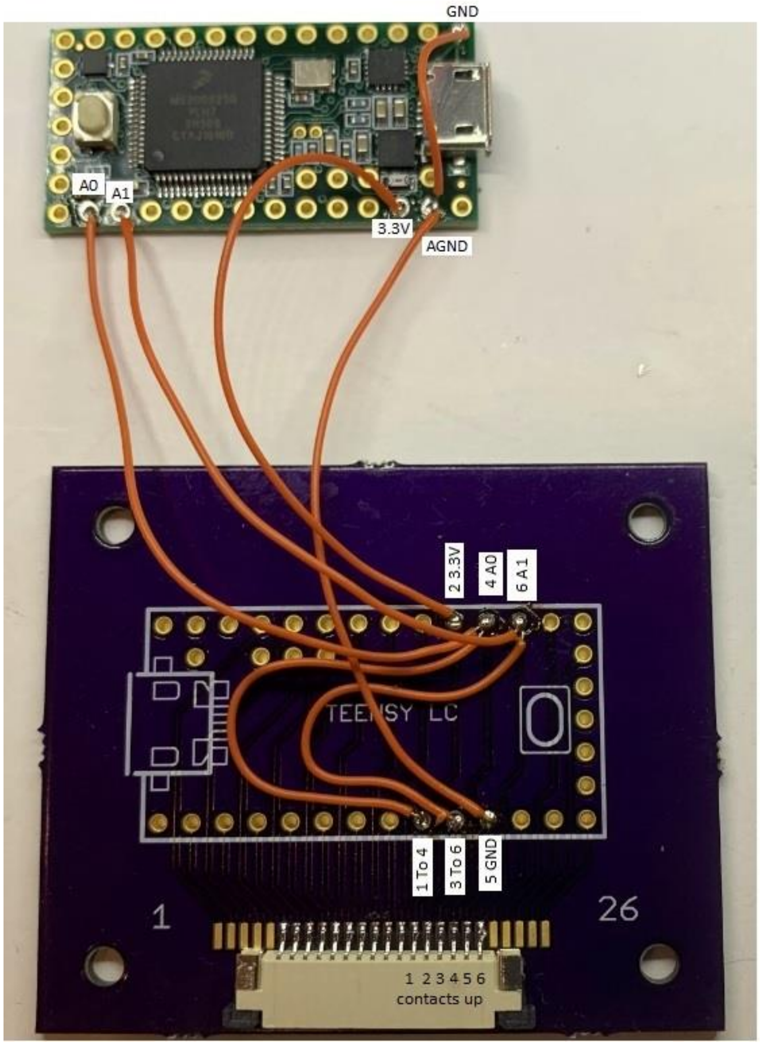
16 pin FPC connector - The first 5 FPC pins are strapped to ground to identify the keyboard as a US language configuration. Later testing showed that pin 6 is for the Fn key. The right and left trackpoint buttons on pins 7 and 8 are pulled up to 5 volts and go to ground when the buttons are pressed. Pins 9 thru 16 are the 8 columns for the key matrix and are pulled up to 5 volts.

18 pin FPC connector - Pins 1 thru 3 and 6 thru 18 are the key matrix rows that are each pulsed low so the columns can be scanned for a closed switch.

6 pin FPC connector - Pin 2 is power and pin 5 is ground. Switching to resistance measurements showed that pins 3 and 6 are tied together and pins 1 and 4 are tied together. These two sets of pins go to the X and Y variable resistors shown below.



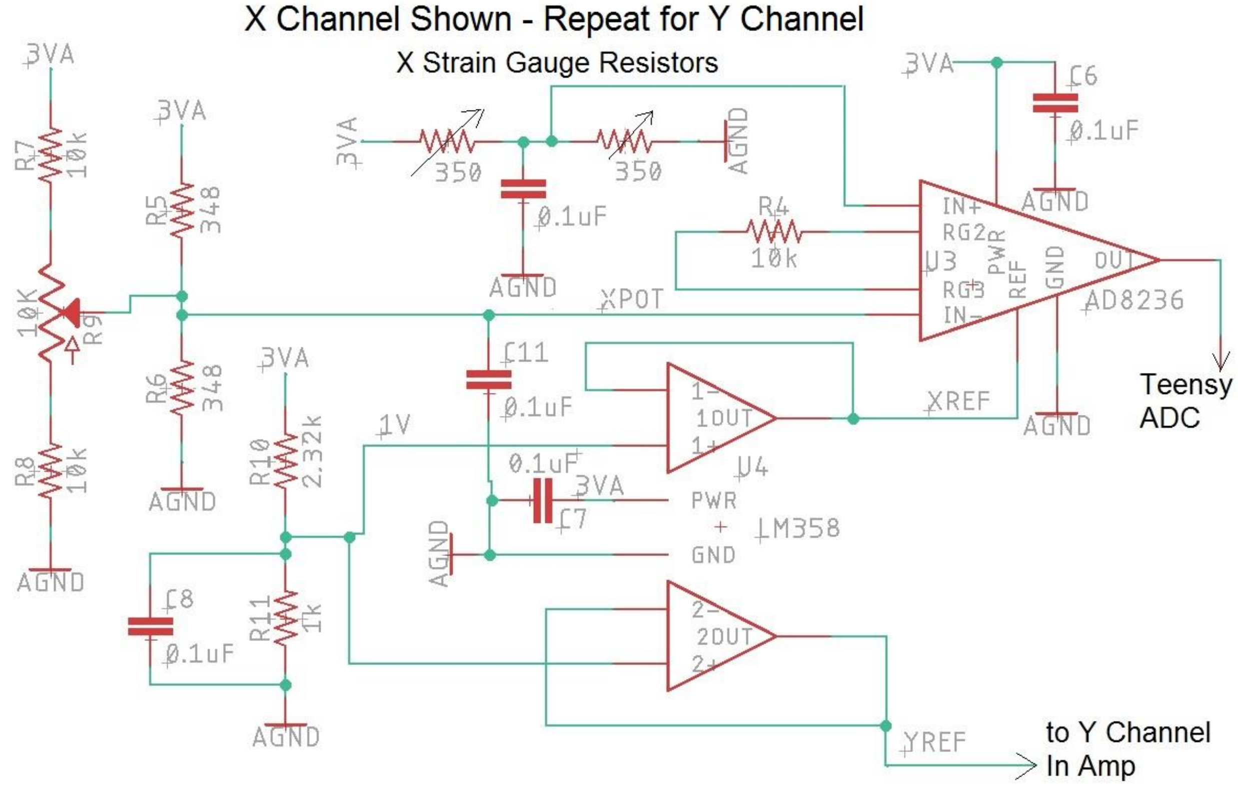
Crude Trackpoint Testing: A crude test board was created with an FPC connector and jumpers to a Teensy 3.2 as shown below.



FPC Pin 2 goes to Teensy 3.3 volts. Pin 5 goes to Teensy ground. Pins 3 and 6 go to Teensy ADC A1 and pins 1 and 4 to Teensy ADC A0.

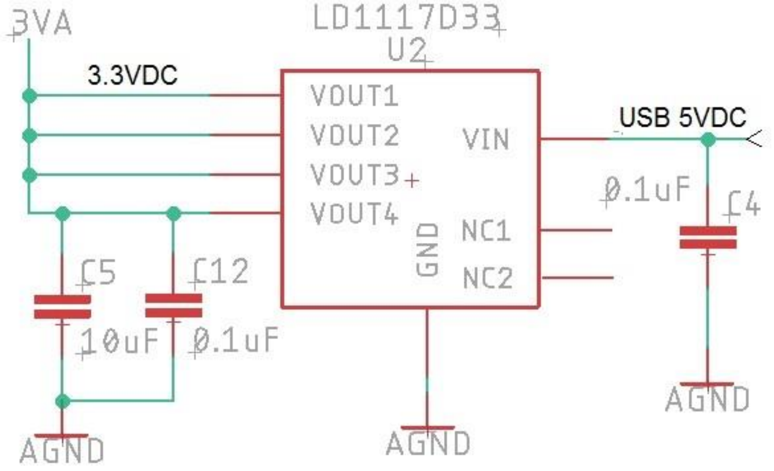
Resistance measurements show the individual strain gauge resistors are 352Ω when at rest. If the trackpoint is forced hard left, right, up, or down, the resistance will change about 1Ω. When powered with 3.3 volts, the voltage divider will only change ± 4mV. I expected this would be difficult to reliably detect with the Teensy but gave it a try anyway. I used code I had previously developed for converting the pointing stick (aka trackpoint) from a Dell D630 to USB. The Dell trackpoint uses 4 KΩ strain gauge resistors that cause a change of ± 30mV which is a lot easier to detect. The code reads the X and Y voltages at startup and saves these as the "at-rest" values. There is a dead zone around the at-rest values so noise doesn't cause unwanted cursor movement. The Thinkpad trackpoint was able to move the cursor with this code but too much force was needed. Reducing the dead zone allowed less force but noise would move the cursor when at rest.

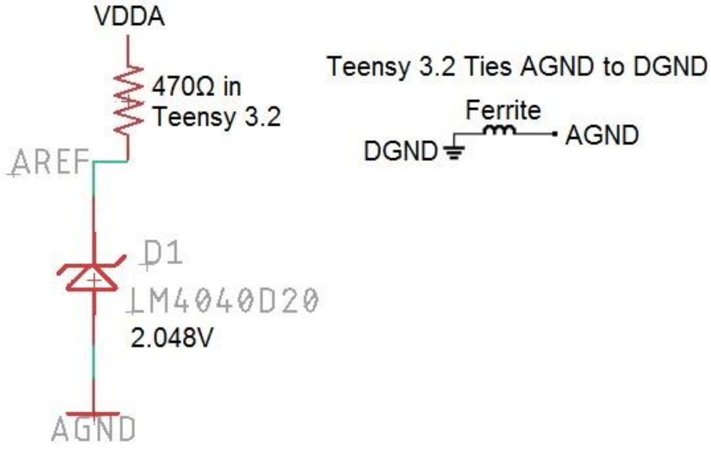
Instrumentation Amplifier: The standard method for reading a strain gauge is to use an instrumentation amplifier. I chose to use the AD8236 because it's low cost, works on a single supply and has a reference pin for offsetting the output. The schematic for the X channel amplifier is shown below. The Y channel is the same. The LM358 Dual Op Amp provides a low impedance 1 volt DC reference voltage to each AD8236. This raises the output up to 1 volt when the inputs are balanced in order to operate from a single 3.3 volt supply. The gain of the AD8236 is set to 47 with a 10KΩ resistor. The 10KΩ potentiometer provides the fine tuning to set the minus input to the exact same voltage as the positive input when the strain gauge is at rest.



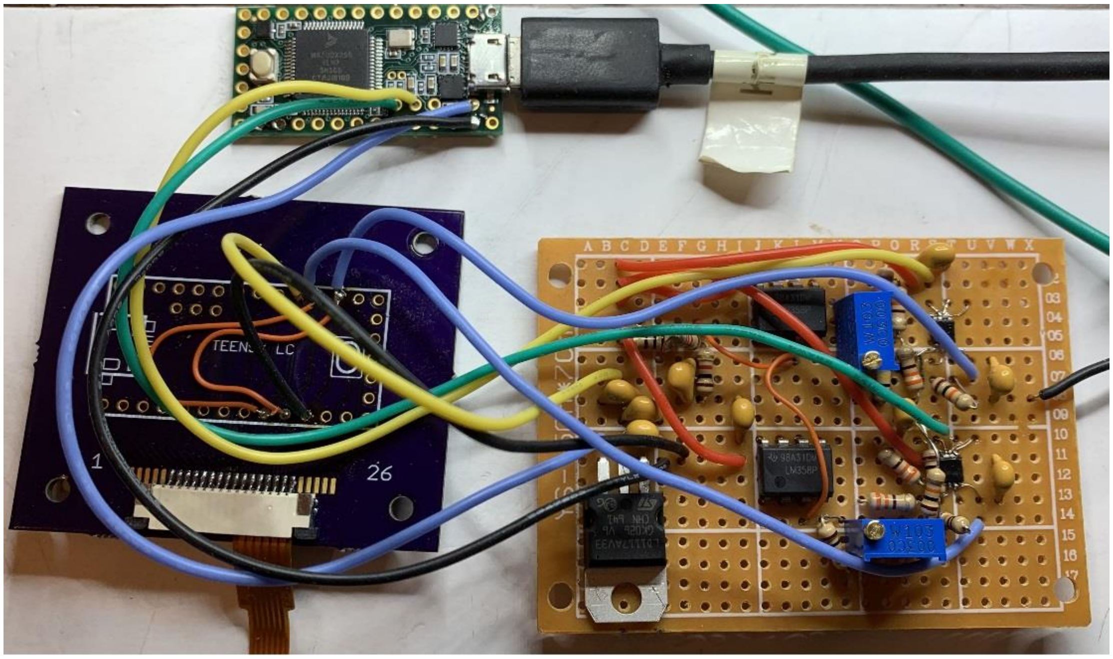
An LM4040D20 shunt style voltage reference is used on the Teensy 3.2 AREF input to give 2.048V for the ADC reference instead of the standard 3.3 volt reference (see schematic below). The 13 usable bits of the Teensy ADC will span 2.048V instead of 3.3 volts. This changes the resolution to 0.25mv/bit instead of 0.4mv/bit.

An LD1117A linear regulator is used to drop the USB 5 volts down to 3.3 volts for powering the trackpoint circuit. This was done instead of using the Teensy's 3.3 volt regulator in order to reduce the digital noise in the amplifiers. Note that the Teensy ties analog ground to digital ground with an inductor as shown below.





Breadboard Circuit: The AD8236 is not available in a DIP but I really wanted to make sure the circuit would work before getting a board fabricated by OSH Park. The breadboard shown below was a test of my soldering skills but after fixing a couple shorts and opens, I managed to get it to work.

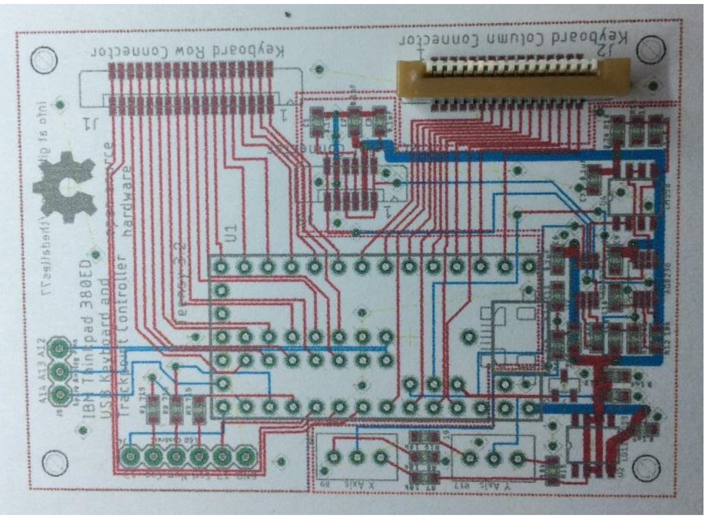


Both potentiometers smoothly adjusted the instrumentation amps to output 1 volt with the trackpoint at rest. Pushing the trackpoint hard in each direction gave these voltages out of each AD8236.

Trackpoint Direction	AD8236 X Channel	AD8236 Y Channel
Right	1.13V	1.0V
Left	0.78V	1.0V
Up	1.0V	1.18V
Down	1.0V	0.84V

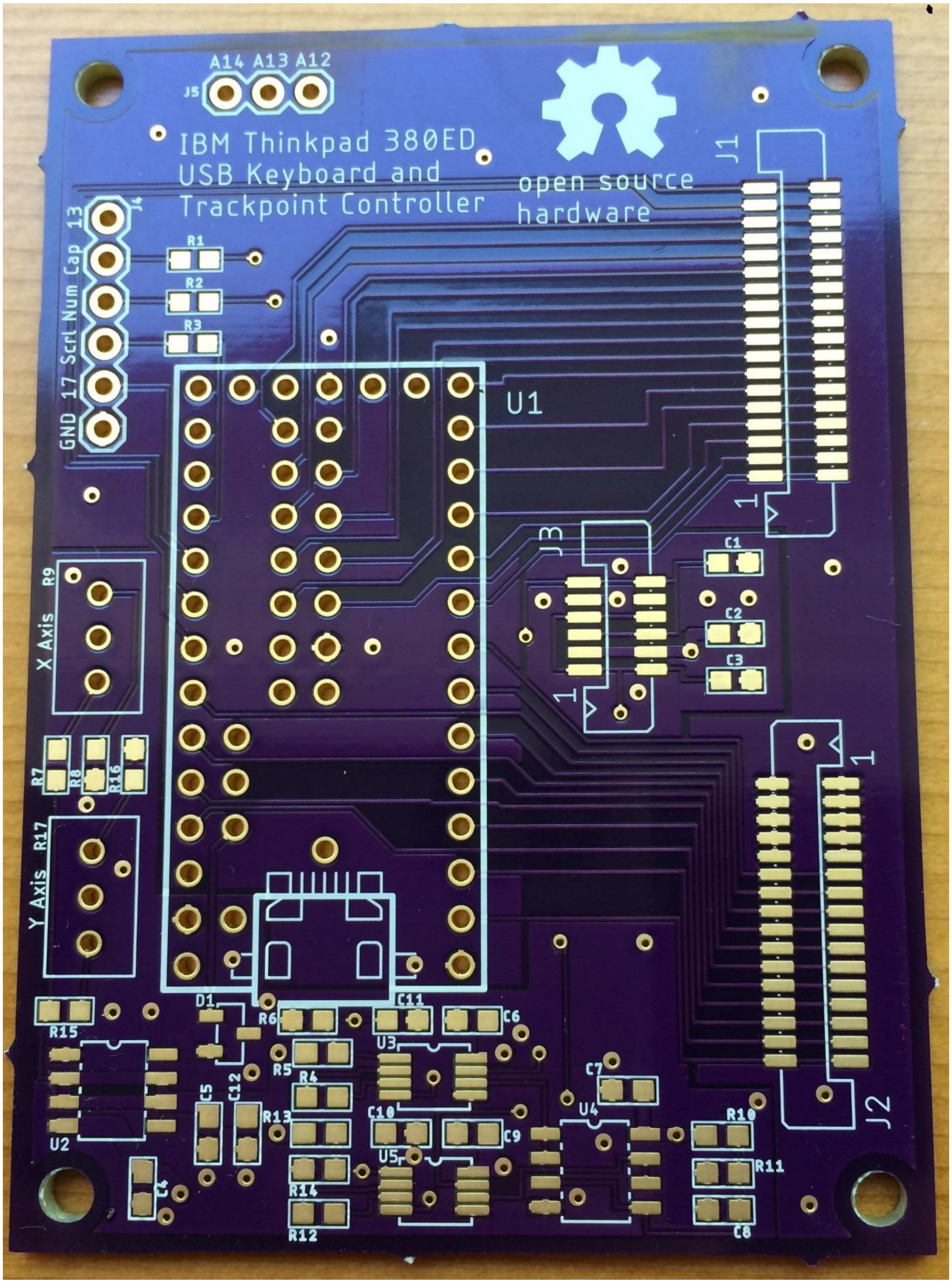
The dead zone in the code was adjusted way down to give a feather light touch on the trackpoint. For temperature testing, I used a heat gun blowing on the keyboard. After a few seconds, the strain gauge resistors deviated slightly which caused a voltage difference on the instrumentation amplifiers. This was gained up to about 10 mv as measured with a meter on the output pin of the amp. This voltage caused the cursor to slowly move up and right until I restarted the Teensy to force it to capture the new at-rest voltage. This problem will be solved in the final version of software by allowing the user to zero the cursor with an Fn-F12 key press. I'll also make it possible for the user to decrease the dead zone using Fn-F9 or increase it with Fn-F10. I'm not a fan of potentiometers and even looked at switching to a Teensy 3.5 to eliminate them. The Teensy 3.5 has 2 DACs which could each be connected thru a resistor to the x and y fixed voltage dividers so that software could nudge the set point up or down. After seeing that the effects of temperature were not very severe, I believe the existing circuit won't drift outside of the 0.2 to 1.8 volt usable ADC range. This range still gives 200mv of head room for normal trackpoint operation.

Eagle Schematic and Layout: I created the footprints and symbols for the components I couldn't easily find online. All the relevant keyboard row and column FPC pins are routed to Teensy I/O pins for maximum flexibility. The general placement of the connectors is driven by the FPC cables. Spare digital and analog I/O are brought to thru-hole pads for future use. 715Ω resistors are included for 2ma Caps, Num, and Scroll Lock LEDs. Area fill polygons are used for analog and digital ground planes on the front and back side. All of the analog components for the trackpoint are located together with the exception of the 6 pin FPC connector. I placed each surface mount part on a 1 to 1 scale paper layout (see below) to verify I had the correct pad size and spacing as well as clearance between parts. I've skipped this step in the past and ended up with placement problems that could have easily been fixed.

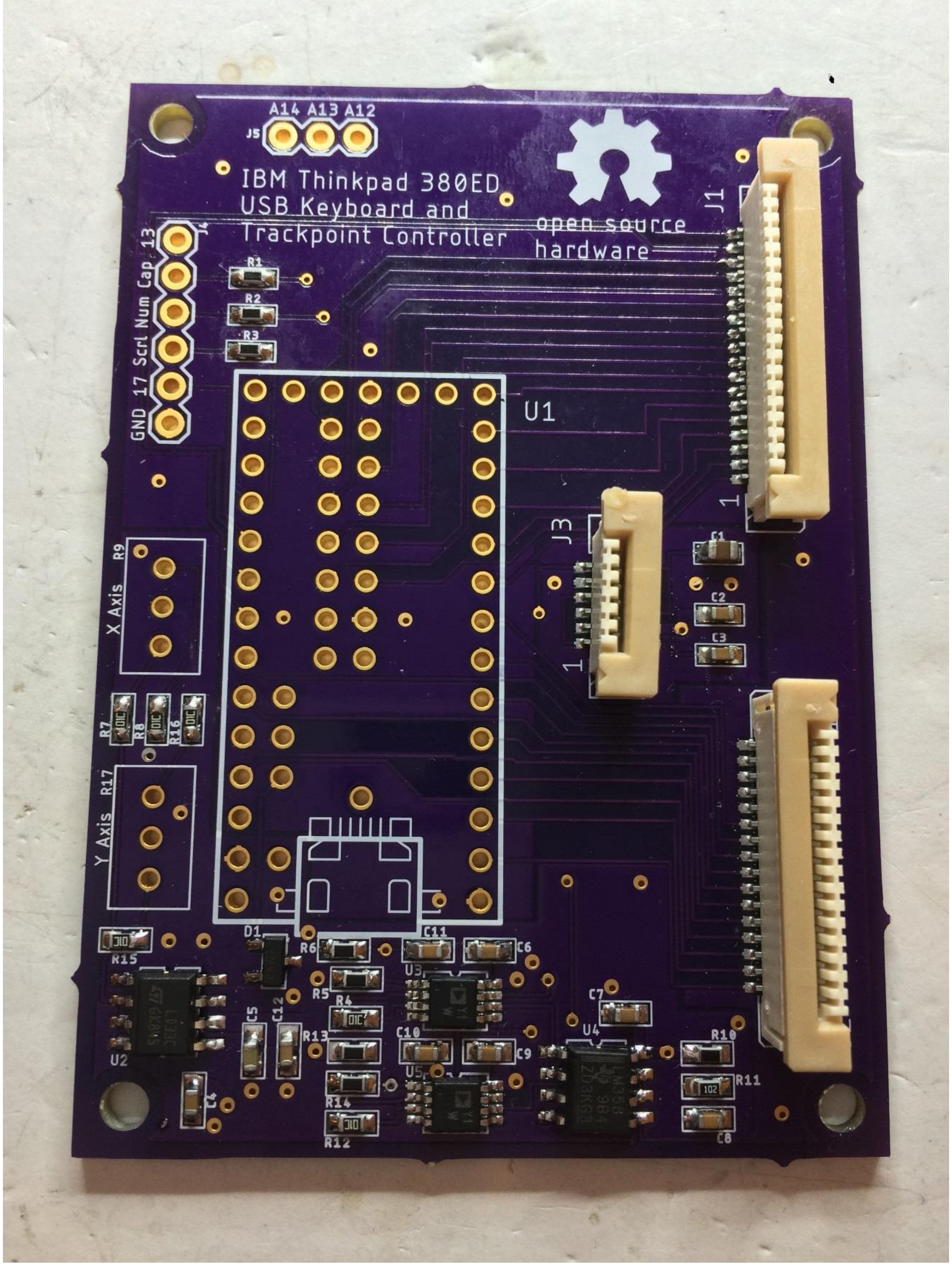


With the paper layout showing no problems, the Eagle board file was sent to OSH Park for fabrication. 3 boards cost $27 to fabricate. The board is 50mm x 70mm and uses the default settings.

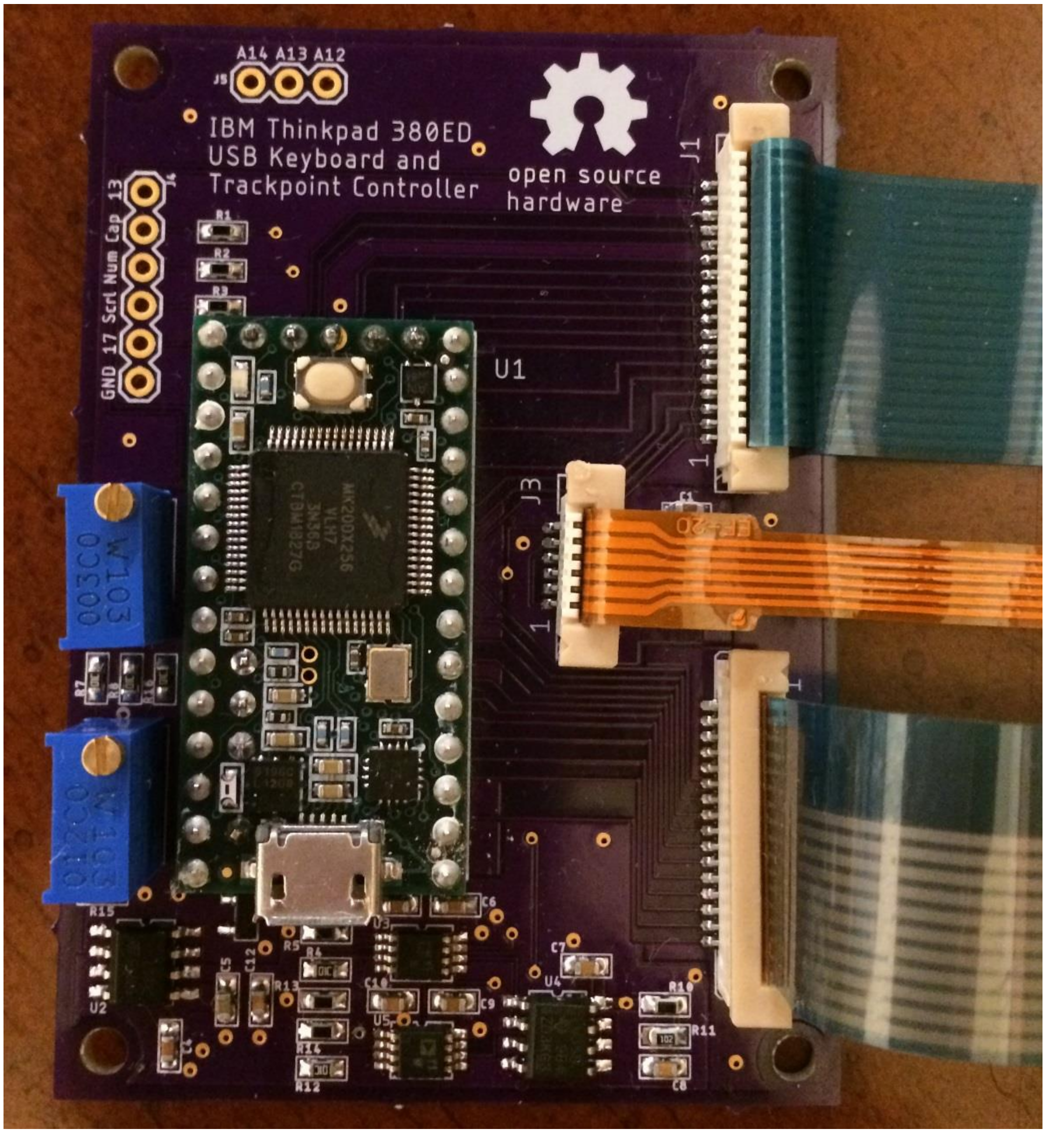
Circuit Board Assembly: The blank OSH Park circuit board is shown below. Wow, they make beautiful boards!



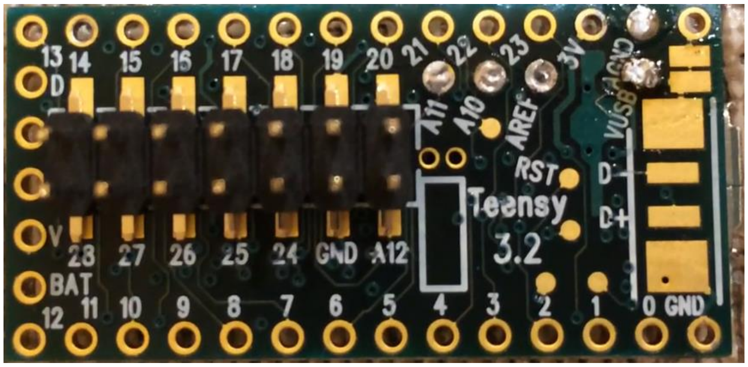
I used standard tin-lead solder paste, applied with a syringe and toothpick, then re-flowed with a converted toaster oven. I had to clear a few solder bridges where I applied too much paste. The old saying "the bigger the blob, the better the job" does not apply with surface mount devices.



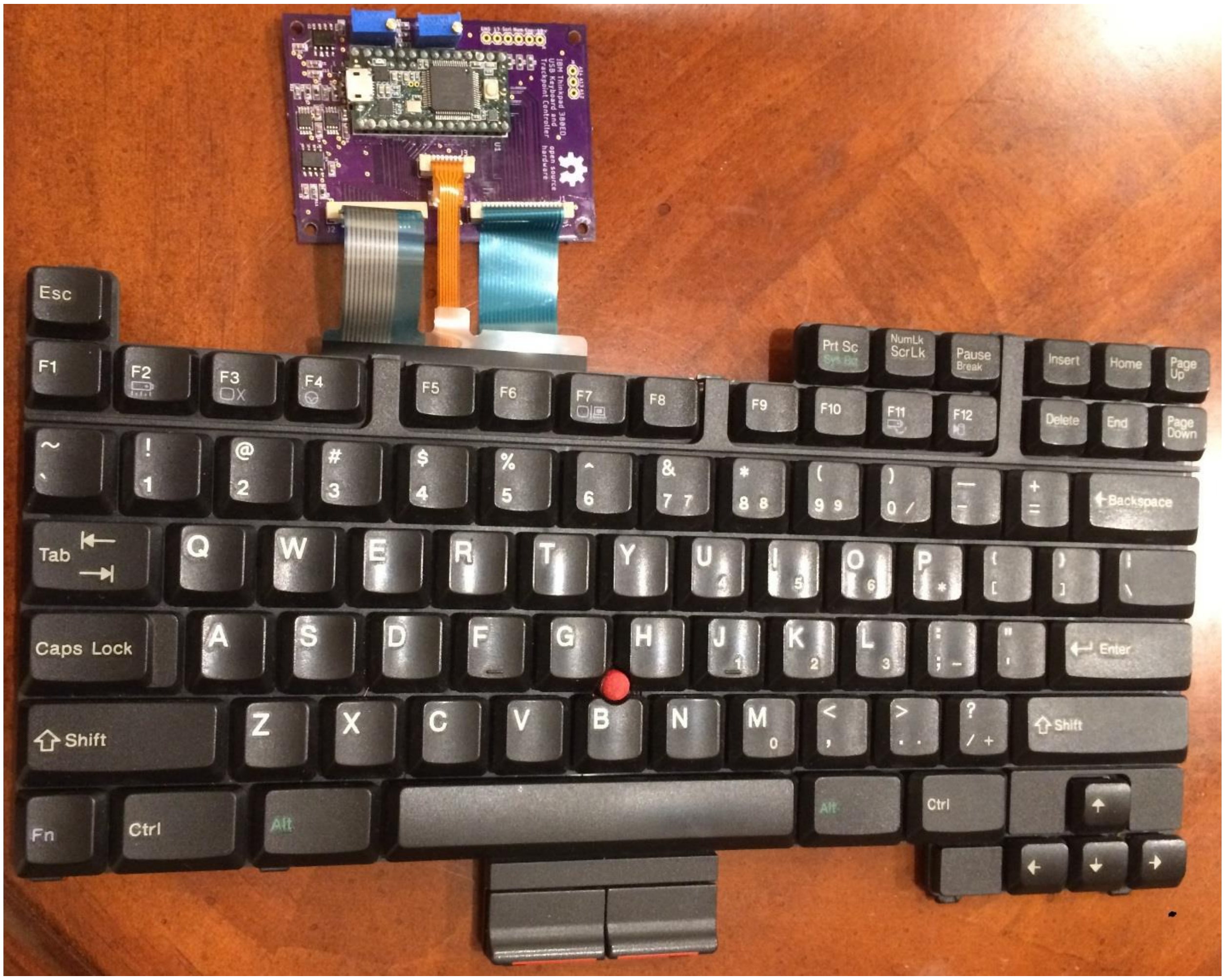
The assembled board with the Teensy 3.2 and FPC cables is shown below.



I designed the layout so that 2x7 right-angle header pins can be used for the surface mount pads on the Teensy backside instead of flying leads.


13 14 15 16 17 18 19 20 21 22 23 3V
D
A11 A10 AREF
VUSB AGND
RST
D-
D+
Teensy
3.2
V
28 27 26 25 24 GND A12
BAT
12 11 10 9 8 7 6 5 4 3 2 1 0 GND

The Keyboard is attached to circuit board as shown below.



Trackpoint Testing: The trackpoint code "Trackpoint_380.ino" from the breadboard circuit was loaded into the Teensy and the potentiometers were adjusted to give 1 volt to the X and Y ADC inputs. The trackpoint operation was identical to the breadboard and worked great. Checking the X and Y ADC voltage a day later showed 0.999V and 1.002V. This voltage will be checked over time to see if it changes.

Key Matrix Decode: The Teensy 3.2 was loaded with the routine "Matrix_Decoder_3p2_380.ino" that scans thru the I/O's and reports over USB, the two pins that are connected when a key is pressed. The resulting I/O connection list (shown below), gives the PJRC key name and the two Teensy I/O's that are connected to that key.

MODIFIERKEY_LEFT_CTRL	2	9
MODIFIERKEY_RIGHT_CTRL	21	9
MODIFIERKEY_LEFT_SHIFT	1	32
MODIFIERKEY_RIGHT_SHIFT	21	32
MODIFIERKEY_LEFT_ALT	0	28
MODIFIERKEY_RIGHT_ALT	4	28
MODIFIERKEY_FN	6	31
KEY_A	22	7
KEY_B	4	10
KEY_C	21	24
KEY_D	22	24
KEY_E	23	24
KEY_F	22	10
KEY_G	0	10
KEY_H	0	11
KEY_I	23	12
KEY_J	22	11
KEY_K	22	12
KEY_L	22	33
KEY_M	21	11
KEY_N	4	11
KEY_O	23	33
KEY_P	23	25
KEY_Q	23	7
KEY_R	23	10
KEY_S	22	8
KEY_T	1	10

KEY_U	23	11
KEY_V	21	10
KEY_W	23	8
KEY_X	21	8
KEY_Y	1	11
KEY_Z	21	7
KEY_TILDE	2	7
KEY_1	3	7
KEY_2	3	8
KEY_3	3	24
KEY_4	3	10
KEY_5	2	10
KEY_6	2	11
KEY_7	3	11
KEY_8	3	12
KEY_9	3	33
KEY_0	3	25
KEY_MINUS	2	25
KEY_EQUAL	2	12
KEY_BACKSPACE	1	26
KEY_ESC	0	7
KEY_F1	2	8
KEY_F2	2	24
KEY_F3	1	24
KEY_F4	0	24
KEY_F5	0	26
KEY_F6	0	12
KEY_F7	1	33
KEY_F8	2	33
KEY_F9	2	26
KEY_F10	3	26
KEY_F11	3	27
KEY_F12	3	29
KEY_INSERT	2	29
KEY_DELETE	2	27

KEY_HOME	2	18
KEY_PAGE_UP	2	19
KEY_PAGE_DOWN	3	19
KEY_END	3	18
KEY_RIGHT	4	29
KEY_LEFT	4	18
KEY_UP	0	18
KEY_DOWN	4	27
KEY_SLASH	4	25
KEY_PERIOD	21	33
KEY_COMMA	21	12
KEY_SEMICOLON	22	25
KEY_QUOTE	0	25
KEY_ENTER	21	26
KEY_LEFT_BRACE	1	25
KEY_RIGHT_BRACE	1	12
KEY_BACKSLASH	22	26
KEY_CAPS_LOCK	1	8
KEY_TAB	1	7
KEY_SPACE	4	26
KEY_PRINTSCREEN	3	28
KEY_NUM_LOCK	23	28
KEY_PAUSE	21	18
LEFT TRACKPOINT	20	30
RIGHT TRACKPOINT	5	30

The connection list was transferred to the keyboard matrix below. The Teensy I/O numbers and FPC connector pins are given for J1 (18 pin FPC connector) and J2 (16 pin FPC connector).

	IO-0/J2-16	IO-1/J2-15	IO-2/J2-13	IO-3/J2-11	IO-4/J2-9	IO-6/J2-6	IO-21/J2-10	IO-22/J2-12	IO-23/J2-14
IO-7 J1-1	ESC	TAB	TILDE	1			Z	A	Q
IO-8 J1-3		CAPSLCK	F1	2			X	S	W
IO-9 J1-4			**LEFT_CTRL**				**RIGHT_CTRL**		
IO-10 J1-5	G	T	5	4	B		V	F	R
IO-11 J1-6	H	Y	6	7	N		M	J	U
IO-12 J1-7	F6	]	EQUAL	8			COMMA	K	i
IO-18 J1-17	UP		HOME	END	LEFT		PAUSE		
IO-19 J1-18			PG UP	PG DN					
IO-24 J1-2	F4	F3	F2	3			C	D	E
IO-25 J1-9	QUOTE	[	MINUS	0	SLASH			SEMICLN	P
IO-26 J1-11	F5	BACKSPCE	F9	F10	SPACE		ENTER	BACKSLSH	
IO-27 J1-12			DELETE	F11	DOWN				
IO-28 J1-13	**LEFT_ALT**			PRNTSCRN	**RIGHT_ALT**				NUMLCK
IO-29 J1-16			INSERT	F12	RIGHT				
IO-31 J1-14						**FN**			
IO-32 J1-10		**LEFT_SHIFT**					**RIGHT_SHIFT**		
IO-33 J1-8		F7	F8	9			PERIOD	L	O

The right mouse button is on I/O-5 (J2-7) and return is on I/O-30 (J1-15).

The left mouse button is I/O-20 (J2-8) and return is on I/O-30 (J1-15).

USB Keyboard and Trackpoint Code: The key matrix table was transferred into the USB keyboard code described in my Instructable. All keys work normally, including the Num-Lock keys. I added mute, volume up, and volume down for Fn-F1, F2, and F3 even though they are not labeled on the keys. The trackpoint code was merged with the keyboard code to give a composite USB device. Function key inputs from the keyboard allow for adjusting the trackpoint variables. Fn-F12 re-captures the "at rest" position in case the user notices any drift. Fn-F9 reduces the noise zone value by 5 and Fn-F10 increases the nose zone by 5. This reduces or increases the amount of pressure needed to move the cursor. Fn-F11 brings the nose zone back to the startup value. The latest keyboard/trackpoint code is available at my GitHub repo along with all the other documentation. The keyboard and trackpoint will continue to be tested and any improvements will be uploaded to my repo.

Stephen designed a stand-alone case that you can 3D print with his files at thingverse.

Parts List:

C1-C4, C6-C12	0.1µF ±20% 16V Ceramic Capacitor X7R 0603	Digikey 399-1099-1-ND	$0.10 x 11
C5	10µF ±10% 10V Ceramic Capacitor X5R 0603	Digikey 490-14372-1-ND	$0.18
D1	LM4040D20IDBZR 2.048V Shunt VRef SOT-23-3	Digikey 296-20899-1-ND	$0.60
J1	FPC Connector 16 pin 1mm pitch Vertical Molex	Digikey WM11009CT-ND	$2.12
J2	FPC Connector 18 pin 1mm pitch Vertical Molex	Digikey WM11011CT-ND	$2.33
J3	FPC Connector 6 pin 1mm pitch Vertical Molex	Digikey WM10993CT-ND	$1.20
R1-R3	715 Ohms ±0.1% 1/10W Chip Resistor 0603	Digikey P715DBCT-ND	$0.35 x 3
R5,R6,R13,R14	348 Ohms ±0.1% 1/10W Chip Resistor 0603	Digikey P348DBCT-ND	$0.34 x 4
R4,R7,R8,R12,R15,R16	10 KOhms ±1% 1/10W Resistor 0603	Digikey 311-10.0KHRCT-ND	$0.10 x 6
R9,R17	10 KOhms 1/2W Thru Hole 25 turn Cermet Pot	Digikey 3296W-103LF-ND	$2.41 x 2
R10	2.32 KOhms ±0.1% 1/10W Chip Resistor 0603	Digikey P2.32KDBCT-ND	$0.34
R11	1 KOhms ±0.1% 1/10W Chip Resistor 0603	Digikey P1.0KDBCT-ND	$0.34
U1	Teensy 3.2 with unsoldered header pins	Digikey 1528-2385-ND	$24.94
U2	LD1117D33CTR Linear Regulator 3.3V 800MA 8-SOIC	Digikey 497-1230-1-ND	$0.53
U3,U5	AD8236 Instrumentation Amp 8MSOP	Digikey AD8236ARMZ-ND	$2.86 x 2
U4	LM358DR General Purpose Dual Op Amp 8-SOIC	Digikey 296-1014-1-ND	$0.36
H1	Teensy Header Kit (instead of 10 flying leads)	Digikey 1568-1422-ND	$1.50
PCB	OSH Park 2 layer 50mm x 70mm circuit board	Eagle board file (lot of 3)	$27.10
			Total = $76.19