Multi-Disciplinary Engineering Design Conference
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York 14623

Project Number: 10003

DYNAMIC KEYBOARD: PHASE II

Adam Stull, EE Student, RIT
Obadiah Pulscher, EE Student, RIT
Joshua Locke, ME Student, RIT
Robert Piccirillo, ME Student, RIT

April 26, 2010

ABSTRACT

The standard keyboard only captures individual binary key stroke events, and thus loses many expressive details in the process. The premise of Phase II of this project, is to create a robust, and consistent device using different key stroke attributes to provide additional functionality to the keyboard. We devised a force sensitive keyboard, that with a learned user skill set can provide the desired additional functionality.

NOMENCLATURE

FSR – Force sensitive resistors. A device in which the normally static resistance value changes in association with changes in pressure applied.

INTRODUCTION

In the original context of this project for Phase I, the scope was defined as:

"Premise: During the act of speaking and signing, the integration of thought and emotion occurs simultaneously and seamlessly with little overt thought. In contrast, text entry differentiates or separates thought and associated emotion. The keyboard only captures key strokes and thus loses emotional expression, even though the act of typing may carry some covert emotions. The goal of this project is to take these learned skills of speaking and signing and use them to enable us to enrich the value of text entry."

This objective proposed that an automated system could be devised to read and differentiate the emotional states of a user by analyzing their typing dynamics. It was found during Phase I of this project however, that due to the high variance and inconsistencies between users, an automated system would not be plausible for simple user use.

For Phase II, the focus has shifted from designing an automated device, to a learned use device. This allows for the user to apply a learned skill set "at will" to manipulate the dynamics of their typing and obtain a desired result. Due to this shift in objective, the project has become more open source and widely applicable while still catering to the original objective.

For Phase III of the project, software development for this open source project will take place. The data provided by the Phase II device will allow for diverse programming applications.

DESIGN PROCESS

Specifications

As defined in the Project Readiness Package[1], the objectives of Phase II of this project are to : integrate the devices and information from Phase I into a full keyboard device and apply PC or controller interfacing to gather data, apply a standard interface to the computer, and define the data integration to allow for programming in Phase III.

The design specifications shown in table (1) were established to meet the current performance parameters of a standard keyboard to ensure broad user compatibility. Specifications for this project were designed with the intent of ensuring its broad user combatility, ease of use, durability, and data integrity.

Metric	Units	Marginal Value	Ideal Value
Sensor Area	cm2	5	>3.64
Max Force Applicable	N	6	10
Req. Cables	Qty	1	2
Response Range	N	0-3	0-10
Powered via USB	mA	500	200
Output Resolution	bits	3	5
Sensor Precision	%E	+/-10	+/-5
Sensor Accuracy	%E	+/-10	+/-5
Min. Duration Detection	ms	30	5
Min. Frequency Detection	ms	5	>10
Keyboard Thickness	Cm	7.5	4
Cable Length	M	1	1.5
Durability	Yrs	1	3
Signal-Noise	Ratio	10:1	100:1
Keyboard Drivers	-	New	Original
Device Weight	lbs	5	2
Cost	$	1100	<900

Table (1) Engineering Specifications

Sensor Selection

Much work was done by the Phase I team in testing and selecting a sensor for this project. The Phase I team selected a FSR, or force sensitive resistor. This device is comprised of two plastic ribbons, with a layer of resistive ink in between. As pressure is applied to the sensor, it's resistive value changes. It is highly durable with virtually no failure points when utilized properly. Due to the low profile nature of the sensor, durability and simplicity of its design, the Phase II team selected to use this sensor as well, however a smaller version of the sensor was used as shown in figure (1). This new sensor with a diameter of .2" allows for a higher percentage of its surface area to be struck by a key stroke, allowing for resistance changes and higher sensitivity.

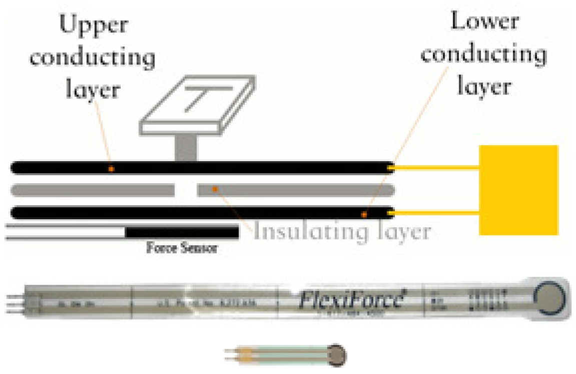


Figure (2) Phase I (middle) and Phase II (bottom) size comparison and sensor placement

Keyboard Selection

The Phase I team opted to redesign the keyboard completely. The Phase II team however, estimated that it would be more efficient, and yield a better final product to modify an existing keyboard to meet the projects needs.

There were several important criteria assessed for the keyboard selection: ergonomics, key cylinder design to ensure a flat consistent contact surface with the sensors, and the ability to integrate with the projects interfacing needs. The keyboard selected is the Sun Microsystems Type 7 keyboard. This keyboard provided an ideal base for the project. The Type 7 interfaces with a PC through USB, which is a widely used, backwards compatible system which is able to provide ~500mA of power to its connected devices. It has ideal key cylinders, that when slightly modified by clipping the side impact braces, provides a clean, level contact surface as can be seen in figure (12). The Type 7 two USB input ports in the bad side of it's shell, as well as a third hidden USB input on the bottom of its shell to allow one periphery device (usually a mouse) to connect through the keyboard to the PC. If utilized correctly, this hidden USB port could be sealed off and if used internally would allow for a controller to communicate to the PC through the same single USB cable. Figure (2) shows the Type 7 keyboard, with a mouse connecting into the hidden USB port on the bottom of the keyboard.

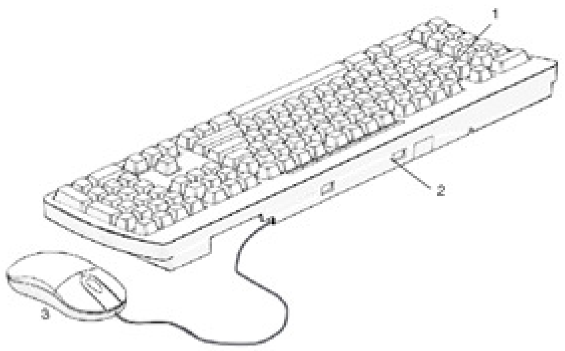


Figure (2) Type 7 keyboard

Sensor Configuration

There are over 70 commonly used character and functions keys on the average keyboard. To allow for analysis of force applied to each key individually, a system had to be devised to simplify the number of information paths required to accomplish this. A total of 75 sensors are required to cover all of the associated typing keys , and every key must be monitored by a standard microcontroller. A “sensor matrix” was derived to allow for a minimal number of required inputs and outputs on a microcontroller. 15 input signals and 5 output signals will be utilized to create a “row-column” array containing all applicable keys. An MM74HC4514N 4-16 IC decoder will be used to further minimize the number of inputs required by providing all 15 input signals (one discarded) in cycle from 4 microcontroller outputs.

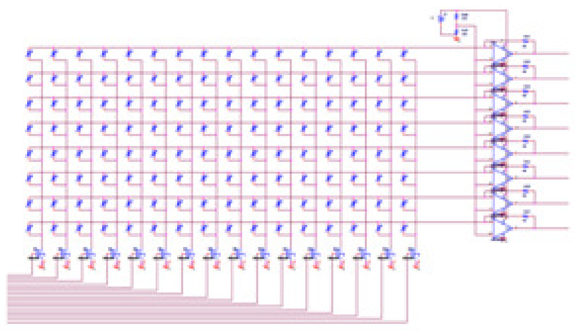


Figure (3) Sensor Matrix

Using this configuration, 4 outputs, and 5 inputs will be required to monitor all 75 sensors. The 5 output signals will be provided by 5 LMC660CN op-amp based amplification circuits, each fed by 15 sensors. A single branch of this circuit simulating (as ground) a single decoder output can be seen in figure (4), the functionality of the “row-column” array can be seen in figure (5), and the original sensor matrix for 148 sensors (which was simplified to 75 by eliminating 3 outputs and 1 input from the design) can be seen in figure (3).

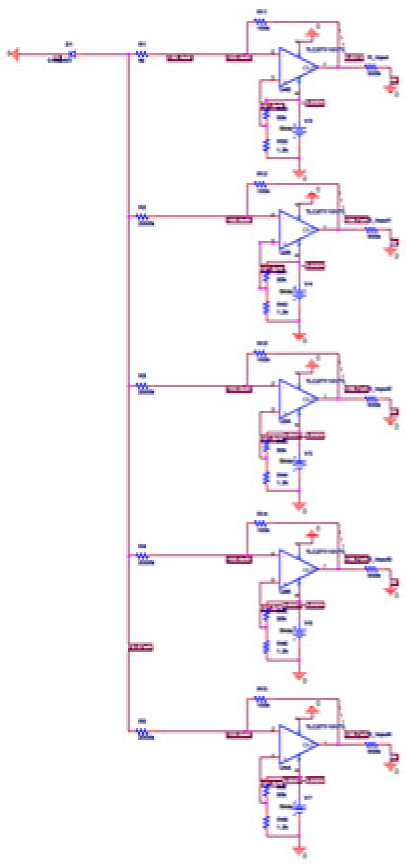


Figure (4) Single Decoder Branch Circuit

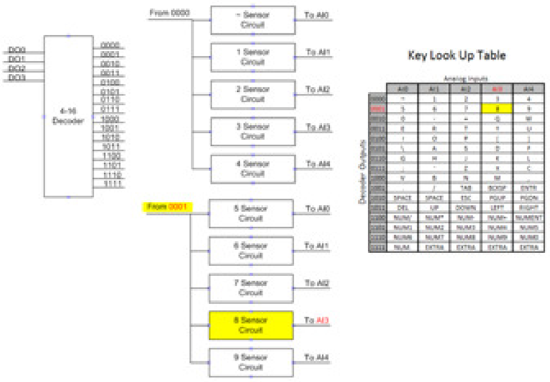


Figure (5) Row-Column Functionality

Device Interfacing

An interfacing network to send the applied force data to the user’s PC is essential in creating a usable open source device. A method for monitoring individual key strikes, storing the data to ensure that it is readily available, and making the data available for interpretation at the PC itself had to be derived.

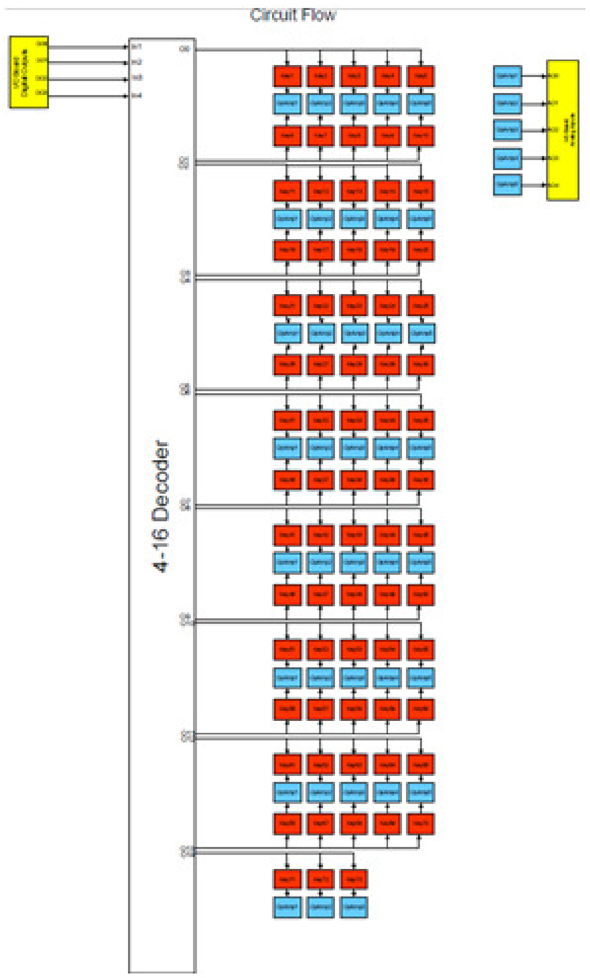


Chart (1) Complete Circuit Flow

A Phidget 1018, as seen in figure (6) microcontroller was selected as the controller device. The Phidget 1018 provided simple USB ready interfacing, as well as 8 digital outputs and 8 analog inputs.

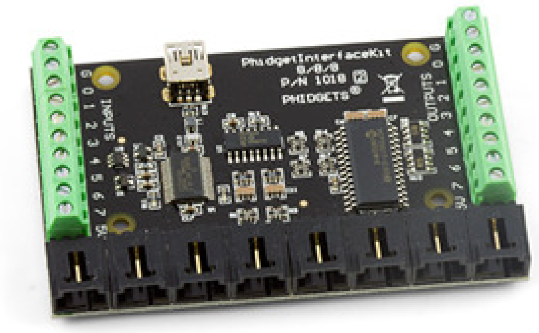


Figure (6) Phidget 1018

However, it was discovered through testing that the analog inputs used to monitor the 5 output amplification circuits, has a sample rate of only 67Hz which was not listed in the device's specifications. This value allowed for a sample rate of ~5Hz. This sample rate is extremely low for our application and as such continued used of this controller was halted. The Phase I group had previously used an Arduino MEGA, shown in figure (7) for their testing purposes, and it was suggested that we test that board for usability. It was found that the Arduino MEGA would allow for a sample rate of 100Hz per key when tested. This value exceeds our design specifications. The Arduino MEGA is similar to the Phidget 1018 in functionality, but provides many more I/O. The Arduino MEGA contains 18 analog inputs, and 32 digital outputs, with a digital output toggle frequency of ~100kHz. The Arduino MEGA provides the same USB capability, though it's faster CPU may result in significantly higher power consumption when assembled and tested.

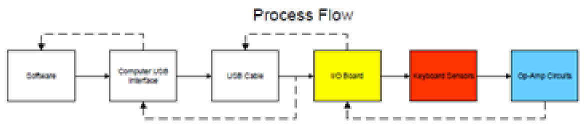


Chart (2) Process Flow

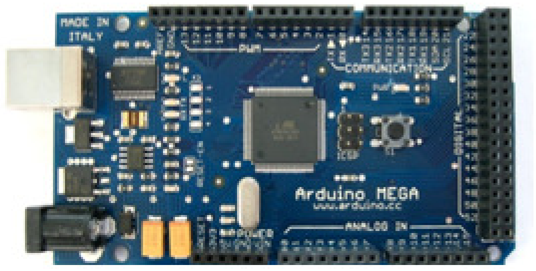


Figure (7) Arduino MEGA

Device Mechanical Feedback

Concerns exist over the linearity of current keyboards. The standard silicone bubble configuration, as seen in figure (8) of keyboards contact membrane requires a load of ~40g before it critically collapses to make contact with the keyboard contact matrix. This required preload has the capability to skew low force data applications, and hence needed to be addressed.

Figure (8) Silicone Bubble Spring

To address this issue, several different types of compressive materials, in various configurations were tested. A Labview application was constructed to provide raw analog data to be used as a comparison. A servo with a rigid arm (Codename: "Smashy") was used to apply a consistent force in a repeatable fashion and eliminate variance in the testing. Human testing was also done as a comparative scope.

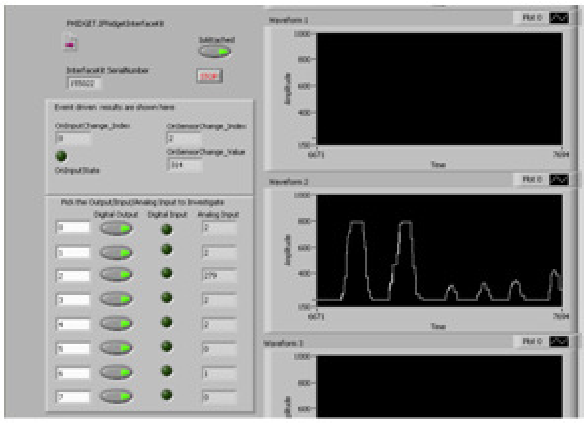


Figure (9) Labview Testing Program

The compressive tested were ¼" red silicone sponge, ¼" cast silicone foam, ¼" diameter cylindrical silicone foam, and a compression spring. These materials were applied inside the key cylinder, and under the key cylinder in varying configurations. The results of which can be seen in figure (11).

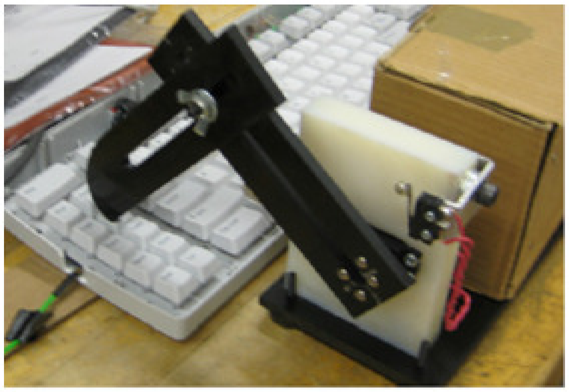
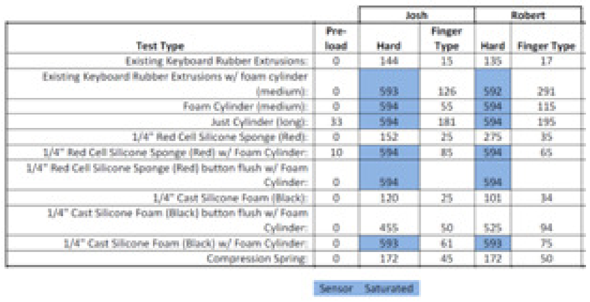


Figure (10) "Smashy"



		Josh		Robert	
Test Type	Pre-load	Hard	Finger Type	Hard	Finger Type
Existing Keyboard Rubber Extrusions:	0	144	15	135	17
Existing Keyboard Rubber Extrusions w/ foam cylinder (medium):	0	🟦 593	126	🟦 592	291
Foam Cylinder (medium):	0	🟦 594	55	🟦 594	115
Just Cylinder (long):	33	🟦 594	181	🟦 594	195
1/4" Red Cell Silicone Sponge (Red):	0	152	25	275	35
1/4" Red Cell Silicone Sponge (Red) w/ Foam Cylinder:	10	🟦 594	85	🟦 594	65
1/4" Red Cell Silicone Sponge (Red) button flush w/ Foam Cylinder:	0	🟦 594		🟦 594	
1/4" Cast Silicone Foam (Black):	0	120	25	101	34
1/4" Cast Silicone Foam (Black) button flush w/ Foam Cylinder:	0	455	50	525	94
1/4" Cast Silicone Foam (Black) w/ Foam Cylinder:	0	🟦 593	61	🟦 593	75
Compression Spring:	0	172	45	172	50

🟦 Sensor Saturated

Figure (11) Compressive Testing Results

The top two performing compressive combinations (Best: foam applied within the key cylinder cut flush with a strip of ¼" black silicon foam as the "spring". Second Best: foam applied within the key cylinder protruding with a strip of ¼" black silicon foam as the "spring") were then applied to human testing. The objective of this test, was to observe the level of approval by actual users to the feel of the keyboard. Linearity, comfort, and overall "feel" were assessed. The human testing participants unanimously chose the foam applied within the key cylinder cut flush with a strip of ¼" black silicon foam as the "spring" configuration as can be seen in figure (12) with and without the foam "spring" layer (left) and a single filled key cylinder (right).

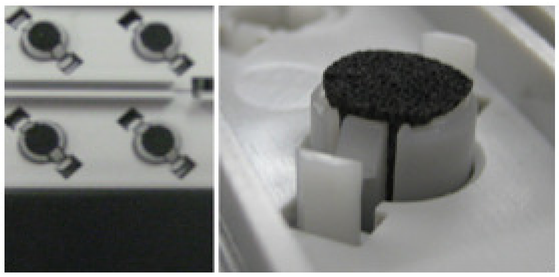


Figure (12) Best Performing Compressive Configuration

PCB DESIGN & FABRICATION

Advanced Circuits PCB artist was used to design the PCB's required for this project. Due to the fabrication PCB width limits, three boards were needed: A sensor layout board for the main body of the keyboard, a sensor layout board for the number pad portion of the keyboard, and a controller interface board to house the amplification circuits. All three boards could be done on two-layer PCB. For cost effectiveness, all three boards were fabricated together, and then cut apart once received.

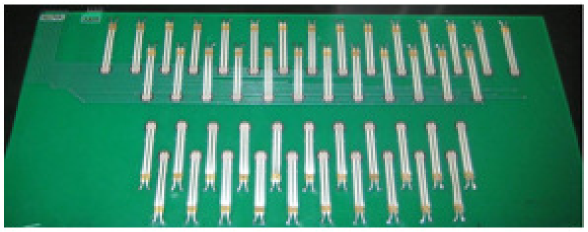


Figure (13) Main Sensor Board

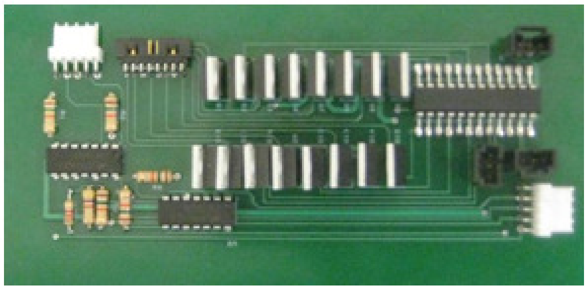


Figure (14) Controller Interface Board

DEVICE PROTOCOL

The Arduino MEGA interfaces with the PC across the USB cable configured to act as a virtual serial port. Upon plugging the system in, the Arduino will begin to repeatedly send the character 'A' until it receives the Character 'Z'. At this point the Arduino considers itself to have connected successfully. To disconnect the device, send the 2 byte pair 'D' 'D' to the device at which point it will return to sending 'A'

When a key is pressed and the force sensitive resistors drop in value causing the output voltage to exceed the threshold. When the Arduino recognizes this change in output voltage it samples and sends the data at a rate of ~60Hz. The data is always transmitted in 2 byte pairs. The first byte is the key number. This is a number between 0 and 79 corresponding to the key which was pressed. The second byte is the force value associated with that key. If multiple keys are pressed at the same time, the data pairs are sent in ascending order by key number.

0	]	20	u	40	N1	60	N7
1	+	21	7	41	up	61	pgup
2	[	22	y	42	down	62	pgdn
3	-	23	6	43	left	63	home
4	p	24	t	44	right	64	end
5	b	25	w	45	unused	65	???
6	g	26	2	46	unused	66	???
7	v	27	q	47	unused	67	???
8	f	28	1	48	unused	68	???
9	c	29	~	49	unused	69	???
10	k	30	???	50	N3	70	/
11	m	31	???	51	N2	71	;
12	j	32	???	52	N5	72	.(period)
13	n	33	???	53	N0	73	L
14	h	34	???	54	N4	74	,(comma)
15	0	35	N6	55	del	75	d
16	o	36	N9	56	ins	76	x
17	9	37	N-	57		77	s
18	1	38	N*	58		78	z
19	8	39	N8	59		79	a

Table (2) Key Number Reference Values

The release of the last key is signaled by sending the 2 byte pair 81-81. If more than one key is pressed, the release can be detected by comparing the key numbers which were present in the previous cycle to the ones in the current cycle. If a key which was present in the previous cycle is not present in the current cycle, that key has been released. Cycles can be distinguished by utilizing the fact that all the key numbers within a current cycle are ascending. A descending key number indicates the start of a new cycle.

The default minimum value for no key press is 51. The threshold can be set by sending the 2 byte pair 'T' followed by the desired threshold. The Arduino will respond by sending the value 'U' and the threshold it set. It is recommended that thresholds above 52 as this is where the circuit biasing causes the value to sit at rest without influence from noise.

447

FINAL TESTING & RESULTS

The compressive materials used in the design were subjected to several tests. Using the finger simulation device, four pounds of force were applied over 25000 times, showing no signs of wear. Strikes against the compressive were also analyzed through high speed video which confirmed the rebound rate of the compressive to ~20Hz, the same as a keyboards' default silicone bubble sheet. As a final compressive test, sample users were allowed to use the device and provide feedback on its overall feel and satisfaction value. Nine out of ten users liked how typing on the keyboard felt and responded to their strikes.

After inserting low-pass filters at the analog inputs of the Arduino MEGA, the measured noise values in the circuitry became 2/200 over an 8 bit output. This gave us a signal to noise ratio of 100:1 , much better than without using the low-pass filters which resulted in a 10:1 ratio. Applying a 100g weight 50 times to the assembled unit yielded a 98% precision rate. The dynamic range of the device was measured to be .1 lbs to 2.64 lbs, or .05kg to 1.2 kg. The measurements through the Arduino MEGA were measured to be at ~60Hz per key, providing a minimum duration detection of 1.6ms. While functioning at full speed, the device required 100mA of power at 5V supplied by the USB port.

All of these tests provided measurements that met or exceeded their respective specifications. The only specifications that were not successfully met or tested were keyboard durability (untested), and moisture resistance (untested).

CONCLUSION

Due to the curvature of the plastic keyboard itself, screws were used to anchor the middle of the PCB, while shims were inserted to curve the PCB and bring its sides up to the height of the plastic.

The 16 pin female connectors used in several locations throughout the keyboard were insufficient. The metal leads easily pulled out from the plastic housing, rendering them useless. Instead the 16 pin connections were hardwired. This was not an ideal situation and new 16 pin connectors should be evaluated.

The decoder layout on the controller PCB was sized incorrectly. The decoder was able to be surface mounted to get around this issue, but the sizing error still exists. The MOSFETs used in the design were also sized incorrectly. Power FETs were used instead of standard sized MOSFETs, taking up more space on the PCB than would be considered ideal. These issues did not affect the functionality, but more the professional quality of the build.

The flat layers of foam were discarded, as they provided too much dampening of the users' applied force. In its place, the cylindrical foam inside each key was elongated by ¼", which then provided the intended functionality.

The overall functionality of the design was tested and verified. After some alterations to the layout of the foam the device worked exactly as expected and exceeded expectations.

PHASE III RECOMMENDATIONS

For design improvements, the issues discussed in the Conclusions section should be taken into account. The overall size of the unit could be reduced, potentially fitting all components into a standard keyboard housing instead of a custom shell. Surface mounted components, smaller MOSFETs, smaller microcontroller, and possibly smaller sensors could be investigated to help reduce the device's overall size.

For the software development, several projects can stem from the development of this device. The first and primary software development can be focused at creating the enhanced typing functionality that the projects original scope and objective for NTID covered. Providing users the ability to toggle between emotional "modes" by the dynamics of their key strikes would provide the functionality needed to satisfy the objective.

Secondly a software package that would allow any user to set/toggle certain functionality in any given program would be ideal. If a user could use intended abnormal striking forces to change fonts, brushes, perform common functions like copy/paste, that would provided a more efficient and diverse keyboard. For example: a programmer setting the Compile function to a hard strike of the 'C' key, or the Save function to a light strike of the 'S' key. A graphic designer could set Grunge Font to a hard strike of the 'G' key, or the Burn Tool function to a light strike of the 'B' key.

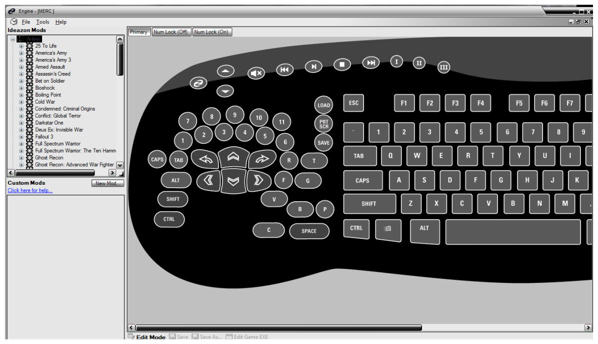


Figure (15) Ideazon Software Package

A software package that provided pre-configured key mappings for commonly used programs that the user could then customize to themselves would be ideal. Figure (15) shows a software package from Ideazon that follows a similar concept. The software automatically toggles between different pre-configured hotkey layouts when it detects certain programs, but the user can also modify the configurations to more suite their personal needs.

REFERENCES:

General
[1] Project Readiness Package P10003: April, 2010
[2]https://edge.rit.edu/content/P10002/public/A%20Guide%20to%20Force%20Sensitive%20Resistors
[3]Labview 8: National Instruments
[4]PCB Artist: Advanced Circuits
[5]Auto-CAD: Autodesk

ACKNOWLEDGMENTS

The dynamic keyboard design team would like to thank Ronald D. Dodge Memorial Endowment Fund for providing us with the funds to research and build a prototype dynamic keyboard. In addition we would like to our faculty guide Prof. George Slack, Dr. Phillips, Dr. DeBartolo, NTID, and Prof. Dino Laury for their participation and assistance in the success of this project.