Smart Chair

Mackenson Jean, Annavay Kean, Bonarine Ramjas, and Thien Nguyen

Dept. of Electrical and Computer Engineering University of Central Florida, Orlando, Florida 32816 – 2450

***Abstract*** – **The project is a smart chair that encourages healthy habits in an office job. One of the many issues facing office workers is back pain caused by long periods of time sitting coupled with poor posture. The product includes an array of sensor that are located on the seat of the office chair and gather data on the user’s sitting trends. The smart chair monitors how long the user sits and reminds them to get up at a regular interval. The device also monitors the user’s posture and provide feedback. The smart chair pairs with an application on the user’s phone to provide feedback and reminders.**

***Index Terms -* Diseases, Internet of Things, Mobile Applications, Pressure Sensors, Web Services**

### I. Introduction

The National Institute of Neurological Disorders and Stroke claims that back pain is the most common job-related disability. Around 80% of adults will experience back pain during their lifetime and 25% of adults have experienced back pain recently.

According to Spine-health’s website, sitting in office chairs involves a ‘static posture’ that can cause or worsen back pain. The increase in back pain is caused by over-stretching the spinal ligaments due to posture used when sitting in an office chair (slouching). A prolonged slouch posture can cause damage to the spine. It is advisable to maintain proper posture when sitting in an office chair to reduce the cause of this pain. Ergonomic chairs can assist in improving back support, but will not stop back pain unless the proper posture is used [1].

According to Spine-health, proper posture includes making sure your office chair is adjusted appropriately as well as your sitting posture.

Regular standing breaks can aid in reducing back pain induced by office chairs. Spine-health recommends getting up from an office chair at least once every half hour for at least one to two minutes.

While the market does offer products to aid in improving posture, those options are very limited and difficult to employ. The design team’s goal was to develop a product that allowed the user to improve their posture without requiring an apparatus to be attached to their body. Our product offers a more-discrete and more comfortable option for users who want to improve their posture.

### II. Overview

The main goal of the project was to use pressure sensors to gather data on weight distribution and to analyze that data to help the user track trends and present the data to the user in a graphical format. In order to meet this objective, the design team elected to use pressure sensors and a modified center-of-gravity equation (with distance averaging) to determine if and in what direction the user was leaning. Once that value is calculated, the data is transmitted to storage in Firebase in the form of a Cartesian coordinate. The user can view this data in the form of a scatter plot whose quadrants correspond to quadrants of the chair (front-left, front-right, rear-left, rear-right). In order to provide more immediate feedback for the user, a vibration feature was added to cause the chair to buzz if the user holds poor posture for a significant period of time.

The posture data is generated by an array of pressure sensors placed on the seat of the chair beneath the foam padding. The placement of these sensors was decided by determining the center of the chair and then identifying where the user’s legs would rest relative to the center.

Storing the user’s posture data is performed by visualizing the chair as a Cartesian grid and determining a variation on center of gravity from the center of the grid. If the user is leaning forward and left, the coordinate will have a negative horizontal (left-to-right) value and a positive vertical (front-to-back) value. Since it is very rare for an individual to sit perfectly dead-center on the chair, a tolerance threshold was introduced to increase the usability of the data.

The vibration feature reminds the user to correct posture by vibrating if the user has held poor posture for a set amount of time. If the user meets this condition, the vibrate modules will vibrate. As some users may find this feature undesirable, the developers added a toggle control to the mobile application to permit the user to turn off the feature.

Storage of the user’s data was performed by way of Google’s Firebase Realtime Database. The embedded software on the WiFi module uploads the Cartesian coordinates to the Firebase database. This data can be accessed via the mobile application.

The mobile application pulls the user’s sensor data from Firebase and displays the coordinate points on a scatter plot. This allows the user to see their sitting trends referenced against the quadrants of the chair.

### III. Power

### IV. Printed Circuit Board

*A. Embedded code logic for operating sensors*

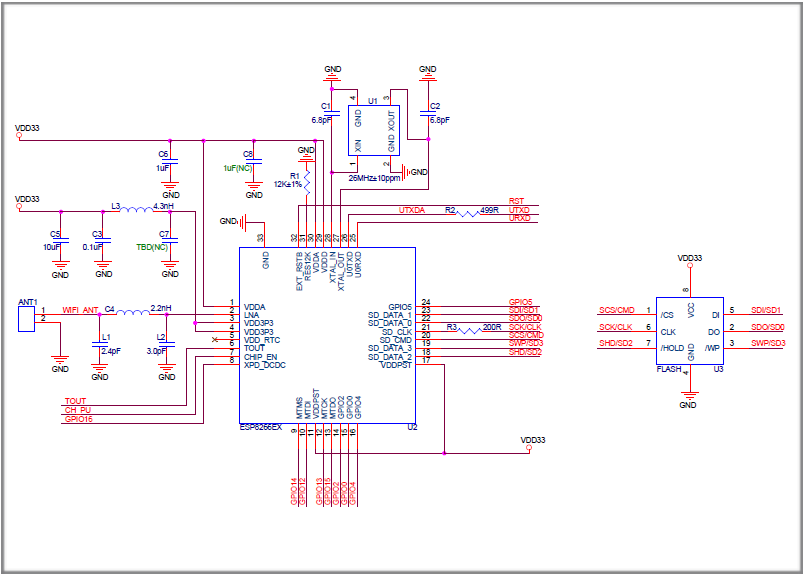
In order to have the capability to connect wirelessly, the device that is used is the NodeMCU using the Wi-Fi capabilities built into the development board. The NodeMCU is a System on Chip (SoC) which is a development board ready to be used along with an MCU to process data and commands, while being simple and easy to work, also inexpensive to acquire with enough pins that comes with analog and digital configurations. This board was impressive due to its capabilities, making the decision to choose it all the more worth while to be implemented for the project.

The NodeMCU is a device that allows Internet of Things platform functionality, such as connecting online via Wi-Fi. The specific module of choice from the NodeMCU lineup, is the ESP8266 that was developed by Espressif Systems that is based in Shanghai. The ESP8266 module itself has different types of variations ranging from the ESP-01 and ESP-12, while all the modules from this lineup has the same processor, what differentiates among them are the breakout board. Wi-Fi modules are important to connecting to the internet, although many equipment on its own are not able to connect to the internet, the capabilities to do so requires having a chip that enables that functionality.

**Pin Configurations:**

* **16 GPIO pins:** PWN pins
  + Control peripherals such as sensors, LEDs, switches, etc.
* **ADC (Analog-to-Digital Converter):** channel access through A0
* **SPI:** 4 pins (SCK, MISO, MOSI, CS)
  + Communication through UART
* **I2C:** accessible support in internal pins
  + Master-Slave device
* **UART:** 2 main interfaces
  + **RXD0 and TXD0:** uploading code into the module
  + **RXD1 and TXD1:** can be used either module or microcontroller

**Specifications:**

* **Operating Voltage:** 5V
* **Flash Memory:** 4Mb
* **Processor:** L106 32-bit
* **CPU Speed:** 80-160 MHz
* **RAM:** 32K + 80K
* **GPIO:** 16
* **ADC:** 1 and 10-bit

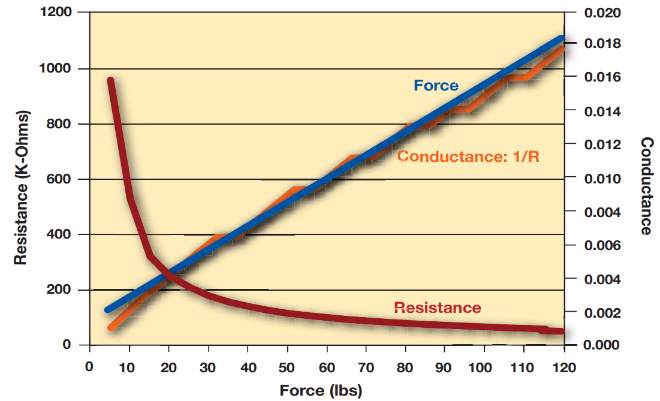
***Schematic of ESP8266***

What makes the ESP8266 important for making the project, is that is allows serial communication between the ATmega2560 MCU to transfer and receive data. This method was proved to be beneficial in the progress of the project, because the data transmission allows being transmitted wirelessly to connect to a database. The database of choice used in the project is Firebase, a mobile and web application that was acquired by Google.

*B. Wireless Communication*

In order to access to the database online, the ESP8266 with its Wi-Fi capabilities has made this possible to have connection to the Firebase database. This makes the process of receiving the data from the ATmega2560 MCU and transmitting sensor values to Firebase. Since the ESP8266 is a microcontroller, it allows compatibility of programs and functions similar to that of the ATmega2560 MCU.

Programming the NodeMCU is almost as simple as the ATmega2560 MCU, while requiring board management and downloading libraries from the manufacturer. The similar compatibility makes it accessible to the various components that can be used as a master and slave device. The transfer of data becomes more seamless thanks to the compatibilities with other MCU and transmitting incoming and outgoing data to the database online via Wi-Fi.

*C. ATMega2560*

Core to the design is the microcontroller which executes the various calculations and communications. The ATMega2560-16AU specifications are summarized in the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Min** | **Typ.** | **Max** |
| Supply Voltage | 4.5v | 5v | 5.5v |
| Clock Frequency |  | 16MHz |  |
| Program Memory Size |  | 256kB |  |
| RAM Size |  | 8kB |  |
| FLASH Size |  | 4kB |  |
| GPIO Pins |  | 86 |  |
| PWM Channels |  |  |  |
| ADC Channels |  | 16 |  |
| ADC Resolution |  | 10 bit |  |
| Interface(s) | 2-wire, UART/USART, SPI | | |

### V. Pressure Sensors

Based on research and specifications of our project as it relates to sensing the user, the sensor type that met our constraints is the piezoresistive force sensor. A brief recap, the piezo sensor applies a known voltage across a variable resistance material, as pressure (or force) is applied to the material, the resistance will change. This change in resistance is not linear across the entire load range of the sensor, however when coupled with the right measuring setup, tuned to a specific range, it can be accurate to within 5 percent. While the sensor cannot accurately determine where within its physical housing the pressure is applied, in an array it can very well offer data on weight distribution relative to other sensors. Considerations made when determining sensor choice include cost, surface area, load range, drive voltage, and response delay. It is important to note that based on the sensor selected, the driving circuit will be constrained to best suited for said sensor. the driving circuit can be a number of choices including inverting/non inverting op amp or voltage divider. In our application, to meet pcb size constraint and cost, we chose the voltage divider setup which allowed for tuning to a range within average weight of a person in the United States. The following chart shows a typical response of resistance vs. force in a piezoresistive force sensor, provided by TEKSCAN.

***Resistance vs. force graph***

In order to verify the functionality of the force sensors, we devised a method to power them and define a reasonable response based on the following parameters: is the return signal amplified, what is the supply voltage, how much force is being applied (this must be some form of standardized weight for calibration. Referencing manufacturer provided test data, we should be able to determine with a high level of certainty, whether the sensors are operating within specification. After verifying functionality to manufacture spec, we confirmed our design theory that the sensors can be amplified using the MCU. We utilized circuit analysis and reference designs to aid in the integration of the crucial components.

In testing the sensors Tekscan recommended the sensor’s be calibrated before initial use. According to their implementation guide, this is done by uniformly over loading the sensors several times; doing so allows sets a reference by which the sensor can be tuned. What is happening during this process is we are looking for a specific voltage output from the sensor based on a ten percent over load (110% loading). Whatever the deviation is from the desired output we can tune the sensor by supply voltage or, line resistor. This line resistor acts as a voltage divider between the supply voltage and the sensor.

Due to the construction of the sensor, it is important that care is taken in mounting the devices. They are designed for point or distributed loading, not shear force measurements which means forces are exclusive to the z-plane we do not want lateral pulling or pushing on the sensor. however, as their name suggests the flexiforce sensor is somewhat malleable which will prove useful for the contours of our chairs seat. Again following the OEM’s recommendation, double stick adhesive tape is used to mount the sensors in place. By using tape versus a permanent adhesive we can ensure that the sensors don’t experience uneven or biased load distribution.

Referenced in the previous section is the concept of load distribution on the sensor. we can load the sensors in a few ways. One involves simply mounting them to the base wood of the chair and then applying the foam padding on top, allowing the foam to act as the buffer and distributor for us. Another method would be to “puck” the sensors; add a hard material onto the sensor’s readable area and let the foam rest upon that. In testing we find that either method is suitable, however since user comfort is important to us, the use of pucks may affect comfort for sensitive users. Based on the test results, we determined that the foam padding of the seat would be a sufficient barrier and provide optimal data, compared to an uncovered sensor in similar test conditions.

*A. Embedded code logic for operating sensors*

Due A key design requirement is to provide the user with meaningful data. This is initiated in the embedded logic of our design. Overlaying a cartesian plane onto our chair’s base, we mapped the outline of our group members’ legs as they sit on the chair. This was combined with available data maps illustrating weight distribution of a seated person of average age and weight, as to not skew the data outside our target demographic. This was translated into an averaging equation from which we calculate an x and y coordinate. The load measured by each sensor scales the result in the corresponding region. Utilizing an interrupt service routine, we are able to define 4 demonstratable conditions that relate to poor seated posture. This interrupt is supported by several functions which allow us to translate the data and send to our database, which can then be retrieved by the user on their android application. The equation utilized follows this format, where n is the number of sensors, A is the known x or y value of that sensor and B is the known ADC output of that same. From this we derive two equations, one for x and one for y.

### VI. Vibration Module

In researching methods of notification commonplace in today’s technology we considered methods employed by smart phones. The vibrate style notification offers discretion and minimal distraction versus sound or light based notifications. Our design called for a small, low power device to be mounted within the chair. We came upon the DC coin vibrate module as a feasible solution. The model in use is the Dowonsol 2v-5v DC Coin Vibration Motor, also known as a pancake coin DC motor. This is small is size measuring at approximately 12mm diameter by 4 mm height. The motor is of a brushed type, and the z axis acceleration measures at approximately 0.8-1.2G. the device can be reflowed for surface mount applications, or can be ordered with an adhesive backing for non-SMT applications. In our design, the drive circuit consists of an NPN BJT, specifically the N2222A. The supply voltage is set at 5v, with peak current consumption at 70mA. From the microcontroller this is controlled with a pulse width modulation (PWM) channel. The circuit is demonstrated below. Nested inside our aforementioned poor posture conditions are triggers for a vibrate sequence. From the user’s phone application we can toggle the modules between on and off.

### VII. Transmission of Data

To have the ESP8266 function with the ATmega2560, Serial Communication is used to establish the connectivity and display the data. The Serial Communication is done so by connecting TX and RX pins as UART interface between both MCU. The starting end must use the TX pin which is connected to RX on the other end and vice versa between the development boards. Ideally the opposite pins must be connected from the boards, one end starts transmitting (TX) to the other board as the receiving end (RX) to establish the communication process.

### VIII. Data Storage

The development team decided to use the Firebase Realtime Database for data storage. Firebase is a NoSQL database that pairs well with the mobile application framework.

After data is transmitted to the database by the embedded software, it is accessed by the mobile application. The data is organized into sets of posture measurements. Each posture measurement is composed of the x and y coordinates as well as a timestamp.

The database also holds a set of flags that are used to communicate settings between the front and back ends. Fields such as ‘notification’ and ‘vibration’ are modified as the user sets user preferences. These flags are checked by both front and back ends to verify that vibrations and notifications should be served. The ‘firePhoneNotif’ field is used by the back end to communicate to the front end that the user is sitting with poor posture and a notification should be served if applicable. The ‘getUp’ field is also set by the back end to enable the front end to serve a reminder for the user to get up and take a walk after a set period of sitting.

The application and the back-end rely on a system of pins to agree on where to read and write data. The pin is hard-coded into the embedded code and the user will received a pamphlet with the product that also encloses the pin. This system works to protect the user’s privacy as user data is stored in a separate database from the account information.

Once a user has created an account, that pin is not longer available for anyone else to use. While the pins should not be duplicated, extra protection is added into the database to prevent writing over another user’s data in the form of the ‘usedPins’ fields. Under the ‘usedPins’ Firebase child, there is a list of pins that have already been claimed by a user. When a user is signing up, if the pin is already claimed, the user is served an error message and the account is not created until a valid pin is entered.

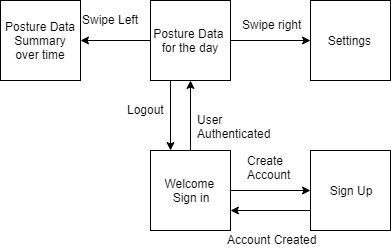
### IX. Mobile Application

A mobile application was deemed the best method to provide meaningful feedback about the posture data to the user. Displaying the sensor data in a graphical format increases the readability and usability of the product.

The mobile application was developed with Google’s Flutter framework in the Dart language. Flutter was selected as the framework for the application due to its reputation for speedy development and compatibility with Firebase.

The application makes use of Firebase’s authentication system to handle user accounts and manage sessions.

The application is composed of 5 screens with distinct functions:

1. Sign In
2. Sign Up
3. Dashboard
4. Long Term Data
5. Settings

The sign in and sign up screens gather and process user input to manage accounts via Firebase.

The dashboard screen is the landing page after the user is authenticated. The dashboard screen displays posture data for the current day on a scatter plot. The dashboard also offers a ‘I am currently experiencing backpain’ button that offers the user a stretch to attempt to help ease the ache. If the user swipes to the right, the long-term data page loads. If the user swipes left, the settings page loads.

The long-term data page offers a scatter plot with all of the user’s data such that the user can use the two plots to track trends and watch their posture improve.

The settings page offers the user a portal to view and update their preferences. Here the user can toggle the ability of the chair to vibrate and the sending of phone notifications. The settings are saved as flag values within the Firebase Realtime Database.

### X. Conclusion

### https://lh3.googleusercontent.com/QVBI1CAqLjsc-gA_qoNGIbosQtqGjarF6DUs4qDLTt2GSEYjpkx1UGX0c5K3Pt3z6XrwAb-vbTqGC1VGStRIrAb9Gb0-KNvy39pVWnP1ZKBO1dVM1opMG3AW4DEZtAvW6lj64w4XI. Biographies

Thien Nguyen is currently a senior at the University of Central Florida and will receive his Bachelor of Science in Computer Engineering. He plans to pursue a career in the field of Embedded Systems.

Annavay Kean is a senior in Computer Engineering at the University of Central Florida. She will be working in software engineering post graduation.

### XII. References