Force Sensitive Dance Pad
Project Proposal

Team:
Sara Akgul - akgul1
Nick Birnberg - birnber2

TA:
Henry Duwe

ECE 445 - Senior Design - Spring 2016
1. Introduction
   1.1 Statement of Purpose
   1.2 Objectives
      1.2.1 Goals and Function
      1.2.2 Benefits and Features
2. Design
   2.1 Block Diagrams
   2.2 Block Descriptions
      2.2.1 Panel Assembly
         2.2.1.1 Panel MCU
         2.2.1.2 Force Sensors
         2.2.1.3 LED Display
         2.2.1.4 Power Supply
      2.2.2 Dance Pad Assembly
         2.2.2.1 Panel Assemblies
         2.2.2.2 Central MCU
         2.2.2.3 Power Supply
      2.2.3 Software
         2.2.3.1 Calibration
         2.2.3.2 Feedback
         2.2.3.3 Driver
3. Block Level Requirements and Verification
   3.1 Requirements & Verification
   3.2 Tolerance Analysis
4. Cost and Schedule
   4.1 Cost
      4.1.1 Labor
      4.1.2 Parts
      4.1.3 Total
   4.2 Schedule
1. Introduction

1.1 Statement of Purpose

Dance Dance Revolution and other dance simulation games are an entertaining way to get active and stay in shape. Beginners and professionals develop hand-eye coordination, rhythm, strength, and endurance with regular play, but only when the dance pad feels like it’s working properly. However, many people still lose interest, and become frustrated. This is because present day DDR pads rely solely on mechanical switches (no other type in production) for input sensing, resulting in poor hardware reliability, inconsistent input sensitivity, and lack of standardization. Our system, based on piezoelectric sensors, would eliminate these problems, integrate easily with current technology, and introduce many new features.

1.2 Objectives

1.2.1 Goals and Function

- Provide consistent, predictable behavior, from pad to pad
- Eliminate the need for constant imprecise non-standardized “mods”
- Reduce player frustration by using sensors that don’t require regular maintenance
- Increase success of competitive events and increase community morale
- Improve the dance game community with a stable standardized platform

1.2.2 Benefits and Features

- Manually adjustable force threshold
- Automatic force threshold calibration by analyzing a player’s style
- Mode for standardized force threshold value -- great for competition
- Easy to install in existing arcade pads and DIY homepad designs
- Works out of box with existing open source Stepmania 5.0 software
- Informative colored LED feedback analogous to force exerted in real time
- Can be used for any dance game with any number of panels due to modularity
- Can be used as generic USB keyboard/ Joystick input
- Feedback based on foot placement triangulation and force data
- Sensor easy auto-calibration
2. Design

2.1. Block Diagrams

![Block Diagram](image)

Figure 1: Panel Assembly

Figure 2: Dance Pad Assembly
2.2. Block Descriptions

2.2.1 Panel Assembly

2.2.1.1 Panel MCU
Directly handles control of the LED Display, decides binary directional triggering in real time based on directions from Central MCU and force profile from force sensor. The panel MCU samples and records the voltages over time of the force sensors, while doing real-time digital signal processing to interpret if the panel was hit within the specified force threshold as well as which color to flash the LEDs. It communicates each hit to the central MCU, while stored voltages are communicated at the end of each song for analytics. It will also be responsible for the initial sampling of the raw force values from the sensor, at about 2kHz, which is sufficient for this application.

2.2.1.2 Force Sensors
We will be using TekScan A201 sensors, which can detect forces from 1-100lb, with the option of changing the range to 1-1000lb if necessary. Typically, the sensor will perform with $\leq 3\%$ linearity error (with line drawn from 0 to 50% load) and repeatability within $\pm 2.5\%$ of Full Scale (Conditioned Sensor, 80% of Full Force Applied). The dynamic range of this force sensor can be modified by changing the drive voltage and adjusting the resistance of the feedback resistor. The latency of sensor response is on average 5 microseconds.

2.2.1.3 LED Display
Multicolored LEDs in pads will light up based on the amount of force exerted on a panel in real-time during play, giving instant feedback to the player. The lights will differ color based upon the amount of force received. The pad will light up green if the player’s force is within the specified force threshold, red if the player is below the threshold, and violet if the player exceeds the threshold. There will also be an option for absolute force colors, to be able to visually compare forces from panel to panel on a single dance pad assembly.

2.2.1.4 Power Supply
Each MCU and associated LEDs and Sensors are connected to a power supply, providing a steady current to power the electronics.
2.2.2 Dance Pad Assembly

2.2.2.1 Panel Assemblies
The dance pad consists of four modular panel assemblies. Each panel assembly operates semi-independently. In theory, we can have any number of panel assemblies connected to a central MCU, depending on how many buttons of a dance game a player wants to use.

2.2.2.2 Central MCU
This microcontroller unit is in charge of issuing control commands to the variable/desired number of panel assemblies, and passing up both raw voltage over time data and directional binary values to software from the panel assemblies up to software for data processing, calibration calculations, and gameplay. USB is used to communicate between the Central MCU and a computer running client software. We need to change the resolution of our data collected by the panel MCU’s to be of a manageable size, yet still providing useful information. We will shrink the resolution of the signals from 1024 bits to 32 bits here. Bottleneck for data throughput will be 2Khz sampling with 32 bit resolution, resulting in a approximately $32 \times 2000 \times 12 \times 180 = 16$MB of raw data on average per song played, which will be easily and quickly transferred over USB for the host to process. USB + Arduino latency is low, and throughput is good, with low latency.

2.2.2.3 Power Supply
The central MCU will also have it's own dedicated power supply, much like the panel MCU's do.

2.2.3. Software

2.2.3.1. Calibration
Initial Calibration sequence for the sensors upon startup of the pad for consistent sensitive for a given set of force threshold values from one physical pad or panel to another. This will ensure that pads loaded with the same settings will always feel the same no matter what.

For our automatic “adjust-to-player” mode, the software will analyze raw force data during play and automatically calibrate the force thresholds by learning the player’s technique and tendencies. This will be done by entering a special software mode, which enables the dance pad to start varying force thresholds an experimentally determine the optimal force values with repeated play and analysis.

Also will allow custom force threshold values to be entered, copied from a past stored configuration, and also “validate” a player’s eligibility for competition on songs completed by checking standardized force thresholds are loaded into the pad.
2.2.3.2. Feedback

The portion of software responsible for feedback will help the user figure out what they can do to improve their playing. We have two main types of feedback that software must handle.

Realtime feedback: LED Feedback - Target Force Window, Absolute Force color gradient

Analytics: Force Triangulation Feedback - Heatmap generation of foot placement, identification of “skill weaknesses” related to specific stepchart “patterns” based on foot placement and force variance during gameplay.

2.2.3.3. Driver

Handles the low level USB communication between the dance pad and PC, communicates sensor threshold mode of operation (loaded force profile, initial calibration mode, adjust-to-player mode), communicates LED data and LED mode of operation (amount of force mode, target force feedback), establishes and manages connection between Central MCU and PC.

3. Block Level Requirements and Verification

3.1. Requirements & Verification

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual Force Sensor</td>
<td>1) - Check voltage to ensure correct voltage input with oscilloscope</td>
</tr>
<tr>
<td></td>
<td>- Probe resistor to ensure proper value for dynamic sensitivity range</td>
</tr>
<tr>
<td></td>
<td>2) - Verify step 1, proper voltage inputs</td>
</tr>
<tr>
<td></td>
<td>- Apply various known weights to sensor: 2lb, 10lb, 50lb, 100lb.</td>
</tr>
<tr>
<td></td>
<td>- Probe voltage output at each weight.</td>
</tr>
<tr>
<td></td>
<td>- Ensure that sensor’s behavior matches voltage distribution on spec sheet.</td>
</tr>
<tr>
<td></td>
<td>3) - Condition the sensor according to the spec sheet, (110lb, 5 times), and power off.</td>
</tr>
<tr>
<td></td>
<td>- Repeat step 2 and record output voltages. Power off and repeat.</td>
</tr>
<tr>
<td><strong>MCUs</strong></td>
<td>- Compute margins of error on repeated trials, and verify under 5%</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| 1) Digitally processes sensor feedback to detect a hit within 10.5 milliseconds.  
2) Receives digital messages containing the force thresholds and LED configurations, setting them accordingly. | 1) - Attach a function generator as an input to the MCU and channel 1 on oscilloscope.  
- Attach output from a GPIO pin to oscilloscope channel 2.  
- Set GPIO pin to high when a hit is detected.  
- Generate an impulse function of the proper threshold voltage.  
- Assure that the pin is set to HIGH within 10.5 milliseconds of the impulse.  
2) - Send specific bit sequence to set force thresholds at 5, 10, 15 lbs.  
- Attach a function generator as an input to the MCU.  
- Attach output from a GPIO pin to a voltmeter.  
- Set GPIO pin to 1 V, 2 V, or 3 V depending on force received and the thresholds.  
- Assure that the proper voltages are set based upon the digital message. |
| **LEDs** | 1) LEDs light up when force threshold is met.  
2) | 1) - Attach a variable power supply as an input to the LEDs unit.  
- Apply three different voltages: 1V, 2V, 3V.  
- Assure that only the red, green, or violet LEDs light up respectively. |
| **Calibration Software** | - Allow configuration of force thresholds.  
2) Load previous configurations. | 1) - Run the client software and click “Configure Force Thresholds”.  
- Assure that the digital message sent matches the one expected and tested for the MCU.  
2) - Run the client software and click “Load Force Thresholds”.  
- Browse for configuration file and click “Upload”.  
- Assure that the digital message sent matches the one expected and tested for the MCU. |
**Feedback Software**
1) Generate visual heatmap of the player’s foot placement.
2) Gives analysis of weaknesses and advice for improvement.

1) - Run the client software and click “Record Moves”.
- Tap the dance pad and record the locations you have tapped visually.
- Click “Stop Recording and Generate Feedback”
- Assure that the displayed graphic matches the locations you have tapped.
2) - Run the client software and click “Record Moves”.
- Tap the dance pad and record the locations you have tapped visually, as well as the relative amount of force used.
- Click “Stop Recording and Generate Feedback”
- Assure that the feedback given for use of force is consistent with the force used on the dance pad.

**Driver**
1) Allows plug & play compatibility to the dance pad.

1) - Connect the dance pad to a PC running the driver.
- Assure that each directional pad is interpreted as arrow keys when pressed upon.
- Assure that pad would work as intended with the game StepMania.

### 3.2. Tolerance Analysis

The most important requirement/block for this project is the Sensor MCU, as it decides at which exact force threshold value to send a directional input to the game. Because we need consistency across different physical dance pads, we need to be able to determine quickly and consistently how to bring a player’s force characteristic data from another setup and make sure it feels exactly the same as another. This means that we must individually test and calibrate all sensors to ensure consistency. By using standard set of weights, we can place 25g weights on each set of sensors, and check their readings to see what voltages are output. We will then pick an offset to normalize each sensor reading so equal amounts of force exerted on each sensor read the same value on the Sensor MCU. Because we strive for consistency, and the Flexiforce sensors we plan to use guarantee less than 3% drift, we will check the voltage outputs from the sensor immediately after calibration, and an hour later, to ensure that we are within this threshold of sensor drift. 3% max drift is acceptable, as this level of sensitivity change in the sensors is FAR better than the drift in mechanical sensors, (which could instantly cease to function entirely due to exposed contacts).
4. Cost and Schedule

4.1 Cost

4.1.1 Labor

<table>
<thead>
<tr>
<th>Name</th>
<th>Hourly Rate (USD/hour)</th>
<th>Hours</th>
<th>Total = ($/hour) x 2.5 x hours to complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nick</td>
<td>$40</td>
<td>250</td>
<td>$25,000</td>
</tr>
<tr>
<td>Sara</td>
<td>$40</td>
<td>250</td>
<td>$25,000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$50,000</td>
</tr>
</tbody>
</table>

4.1.2 Parts

<table>
<thead>
<tr>
<th>Part</th>
<th>Unit Cost</th>
<th>Number</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino Uno R3</td>
<td>$25</td>
<td>5</td>
<td>$125</td>
</tr>
<tr>
<td>TekScan A201</td>
<td>$14.63</td>
<td>12</td>
<td>$175.50</td>
</tr>
<tr>
<td>Red LED</td>
<td>$0.32</td>
<td>32</td>
<td>$10.24</td>
</tr>
<tr>
<td>Green LED</td>
<td>$0.32</td>
<td>32</td>
<td>$10.24</td>
</tr>
<tr>
<td>Violet LED</td>
<td>$0.32</td>
<td>32</td>
<td>$10.24</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$331.22</td>
</tr>
</tbody>
</table>

4.1.3 Total

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Labor</td>
<td>$50,000</td>
</tr>
<tr>
<td>Parts</td>
<td>$331.22</td>
</tr>
<tr>
<td></td>
<td>$50,331.22</td>
</tr>
</tbody>
</table>
### 4.2 Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Task</th>
<th>Assignee</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/8</td>
<td>Finish Design Proposal</td>
<td>NB, SA</td>
</tr>
<tr>
<td>2/15</td>
<td>Design Review</td>
<td>NB, SA</td>
</tr>
<tr>
<td>2/22</td>
<td>Breadboard a Panel Assembly</td>
<td>SA</td>
</tr>
<tr>
<td></td>
<td>Finish Linux Plug &amp; Play Driver For One Panel</td>
<td>NB</td>
</tr>
<tr>
<td>2/29</td>
<td>PCB Design and Manufacture</td>
<td>NB, SA</td>
</tr>
<tr>
<td>3/7</td>
<td>Complete LED feedback</td>
<td>NB</td>
</tr>
<tr>
<td></td>
<td>PCB Assembly</td>
<td>SA</td>
</tr>
<tr>
<td>3/14</td>
<td>Finish and Test Single Panel Assembly</td>
<td>SA</td>
</tr>
<tr>
<td></td>
<td>Finish and Test Basic Analytics Software</td>
<td>NB</td>
</tr>
<tr>
<td>3/21</td>
<td><em>Company Retreat</em></td>
<td>NB, SA</td>
</tr>
<tr>
<td>3/28</td>
<td>Finish Driver for Dance Pad</td>
<td>SA</td>
</tr>
<tr>
<td></td>
<td>Finish Analytics Software</td>
<td>NB</td>
</tr>
<tr>
<td>4/4</td>
<td>Finish Calibration Software</td>
<td>SA</td>
</tr>
<tr>
<td></td>
<td>Test and Calibrate Sensors</td>
<td>NB</td>
</tr>
<tr>
<td>4/11</td>
<td>Focus on User Experience on Client Software</td>
<td>NB</td>
</tr>
<tr>
<td></td>
<td>Integrate Circuitoery With Dance Pad Hardware</td>
<td>SA</td>
</tr>
<tr>
<td>4/18</td>
<td>Testing/Debugging</td>
<td>NB, SA</td>
</tr>
<tr>
<td>Date</td>
<td>Activity</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>----------</td>
<td>-------</td>
</tr>
<tr>
<td>4/25</td>
<td>Demonstrations + Presentations</td>
<td>NB, SA</td>
</tr>
<tr>
<td>5/2</td>
<td>Demonstrations + Presentations</td>
<td>NB, SA</td>
</tr>
</tbody>
</table>