**Project Title:** Pressure Run

**Name:** Ta'shawnna Green

**Introduction:**

The goal of this project, Pressure Run, was to design a fun, sensor-based challenge, interactive two-player maze navigation game that integrates real-world sensor inputs to control on screen gameplay. This project combines hardware and software to create a dynamic, competitive experience where players physically interact with the system using a force-sensitive resistor (FSR) and an accelerometer. The FSR controls horizontal movement (rightward), while the accelerometer controls vertical (up and down) movement based on tilt. Players navigate through a custom-designed maze, attempting to reach a finish line (dark green dot) as quickly as possible without crashing into walls.

The user experience is highly interactive in which they hold the 3D printed joystick with the embedded accelerometer and the 3D printed push button which holds the FSR. These sensors are connected through a breadboard circuit. Each player gets one attempt, and if they crash, their distance is recorded. If they successfully reach the goal, their completion time is captured. After both players have finished, the program automatically determines the winner based on who finished first or who traveled farther if neither finish. To reflect our UNC-Chapel Hill spirit, the maze background was styled Carolina blue, the finish zone was represented as a dark green dot, and the maze walls were colored white.

I also added celebratory effects such as "Congratulations!" banner and a confetti animation that rains down after the winner is announced, enhancing the excitement and visual appeal of the game. The use of real-time data acquisition (DAQ) combined with a fully customized MATLAB App Designer interface allowed us to blend hardware, signal processing, and real-time visualization into the final product. Pressure Run not only demonstrated sensor integration and circuit design but also showcased thoughtful user interface design and system responsiveness, resulting in an engaging experience.

**Sensor design:**
The system utilizes two analog sensors: a SparkFun small FSR and an ADXL337 3-axis accelerometer. These sensors allow real-time user interaction with the game by translating physical input (force and tilt) into analog electrical signals. Below, we detail how each sensor functions and how its specifications aligned with the goals of our game design.

**Force Sensitive Resistor (FSR)**

***Figure 1:*** This FSR has a 4 mm (0.16'') active sensing area and operates on a simple resistive principle: its resistance decreases as pressure increases. When no pressure is applied, the resistance is greater than 1 MΩ. When fully pressed, the resistance can drop to approximately 2.5 kΩ. We incorporated this sensor into a voltage divider circuit, with a fixed 10 kΩ resistor and the divided voltage was connected to analog input 1+ of the AD3. As pressure is applied to the FSR (using a 3D-printed thumb button), the voltage across the fixed resistor increases and is read into the analog input of our AD3 data acquisition (DAQ) device, which was configured to 3V. When the voltage surpassed a preset threshold, the game character would move to the right on the x-axis. While FSRs are not highly precise, they are more than adequate for discrete user input detection and offered a force sensitivity range of 0.1 N to 10 N and low actuation threshold (2 g) which makes it ideal for simple user input detection. The FSR's high resistance range made it sensitive enough to detect gentle presses.

**ADXL337 Accelerometer**


***Figure 2:*** The ADXL337 is a small, low-power, 3-axis analog output accelerometer with a full-scale range of ±3 g. It operates on a 3.3V supply and outputs a resting voltage of ~1.65V per axis (for 0 g), but we operated on 3V from the AD3. The output is proportional to tilt or acceleration on each axis, which we used to control vertical movement in our game. The sensor outputs an analog voltage for each axis (X, Y, Z) corresponding to measured acceleration. In our design, only the Y-axis signal was used. The accelerometer was powered via the 3.3V rail and GND connecting to there respective ground and power on the breadboards AD3 ground and power connections while the Y-axis voltage was read using analog input 2+. When the player tilted the 3D-printed joystick, the voltage output changed from a baseline of 1.5V (corresponding to 0g). Positive tilt resulted in voltage >1.5V (upward movement of character), and negative tilt yielded voltage <1.5V (downward movement of character). To minimize electrical noise and false movements, we passed this signal through a low-pass RC filter, with a cutoff frequency chosen to balance responsiveness and stability. The ADXL337 offered reliable tilt sensing within a ±3g range with about 330 mV/g sensitivity.

**Circuit design:**

**Final Parts List**

SparkFun Small Circular FSR, SparkFun ADXL337 Accelerometer, 10 kΩ resistor, 1kΩ resistor, 0.1 µF capacitor (for RC filtering), AD3 data acquisition device (for analog input), Custom 3D-printed FSR button, Joystick mount for accelerometer, Breadboard and wires

**Circuit Diagrams**

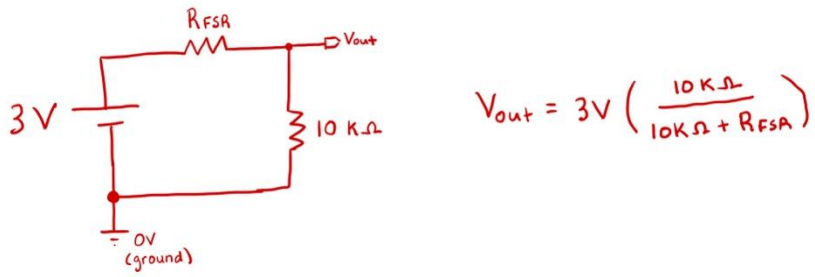



***Figure 3:*** This is the voltage divider for the FSR along with the equation so that Vout increases. When no force is applied, the FSR has high resistance. Here RFSR decreases with more force, and Vout increases because the denominator gets smaller, resulting in a higher output voltage. One leg of the FSR goes to 3V and the other leg goes to the 10 kΩ resistor that is connected to the ground. The voltage node between FSR and resistor is Vout and Vout is sent to the 1+ of the AD3.

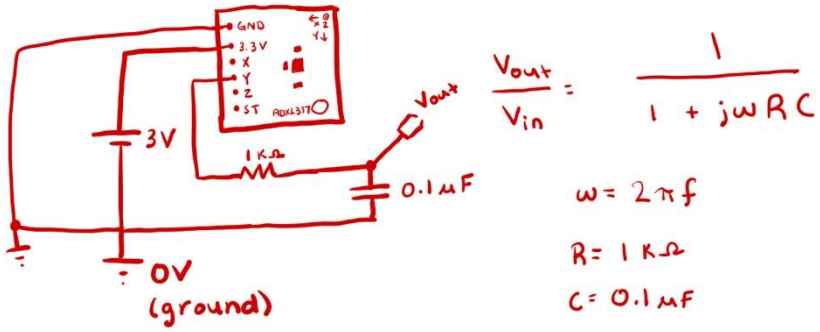



***Figure 4:*** This is the circuit diagram for the accelerometer connections including an RC low pass filter along with the equation for Vout/Vin in the phasor domain. There is a capacitor that connects from after the resistor to the ground. The voltage across the capacitor is Vout, and it is connected to 2+ of the AD3. The filter reduces high-frequency noise from the accelerometer with a cutoff frequency of 159 Hz.

**Typical Output Values**

FSR Output from the SparkFun datasheet:

- At no pressure, Rfsr > 1 MΩ → Vout is approximately 0.03 V
- At firm pressure, Rfsr is approximately 2.5 kΩ →$Vout = 3 * \left(\frac{10000}{10000+2500}\right) = 2.4\ V$
- Output ranges from 0 V to 2.4 V

Accelerometer Output:

- The ADXL337 outputs 1.65 V at 0g when powered by 3.3 V
- For ±3g range: ±0.33 V swing (sensitivity ≈ 330 mV/g)
- Typical range on Y: If tilted or moved: ~1.3 V (−1g) to 2.0 V (+1g)

**Cutoff Frequency Selection**

$$f_c = \frac{1}{2\pi RC} = \frac{1}{2\pi (1\,k\Omega)(0.1\,\mu F)} = 1592\ Hz$$

***Figure 5:*** To reduce high-frequency noise from the ADXL337 accelerometer, we used a first-order RC low-pass filter on the Y-axis signal before it reached the AD3 (2+). With a ~1.6 kHz cutoff, the filter preserves motion signals under 500 Hz while smoothing out spikes. This balances noise reduction with fast response, improving real-time control without introducing delay.

**Design Choices and Tradeoffs**

Maximum Sensitivity:

- The FSR was used in a voltage divider with a 10 kΩ resistor to create a wide voltage swing over its pressure range (from <0.1 V to ~2.4 V), maximizing detection of even small pressure differences.
- The RC filter on the Y-output of the accelerometer helped reduce jitter/noise, making movement tracking smoother for the game character.

Tradeoffs:

- The FSR is not linear and lacks precision, but it was good enough for detecting pressure-based triggers.
- The RC filter could slightly delay signal response due to its time constant but was acceptable since high-speed movement wasn't needed for our joystick input.

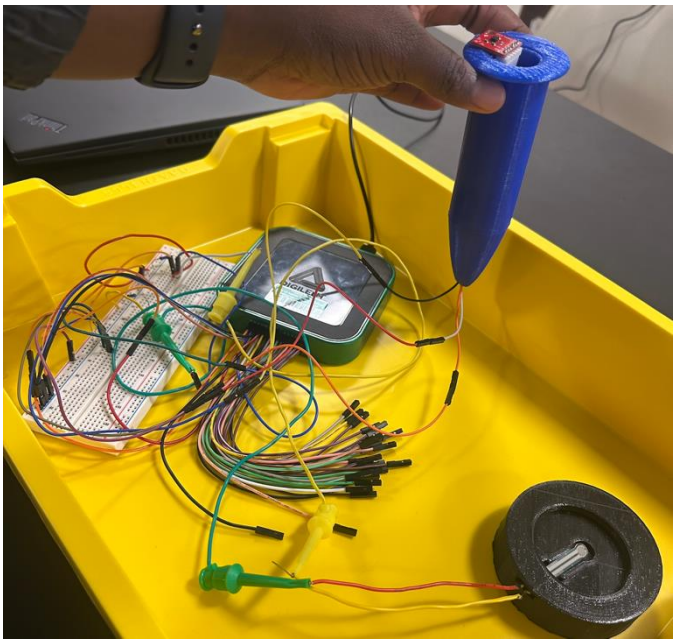
**Program design:**

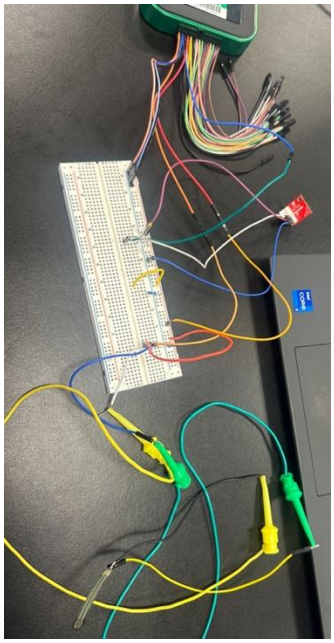
The MATLAB program for our game, Pressure Run, was developed using MATLAB App Designer, a platform we had not used previously in class but found to be incredibly helpful for building user-friendly graphical interfaces. The App Designer allowed us to create buttons, axes, and game logic in an organized structure. Our app consists of a single screen with a UIAxes panel that displays the game field, and a "Start Game" button that initializes the game. When clicked, the app powers on the sensors, sets up the circuit connections via the AD3 DAQ interface, and runs a continuous game loop.

During the game, a red or blue dot (representing Player 1 or Player 2) moves across a maze using real-time sensor input. The FSR (Force Sensitive Resistor) controls rightward X-axis motion when pressed, and the ADXL337 accelerometer controls vertical movement through changes in Y-axis voltage. We set up live analog reads from the FSR and the accelerometer's Y-axis through channels 1+ and 2+ on the AD3, respectively. The game processes those voltages and translates them into dot movement on the screen. Maze walls are drawn using rectangle() objects, and collision detection checks whether the player intersects a wall or reaches the green finish line. After each turn, the code logs whether the player finished or crashed, and compares the two runs to announce a winner or declare a tie. Confetti animation and congratulations messages make the ending celebratory and interactive.

The program features automatic DAQ resetting between players to prevent data communication errors, a customized Carolina Blue background to match our UNC school colors, and enhanced visual celebrations at the end of the game.

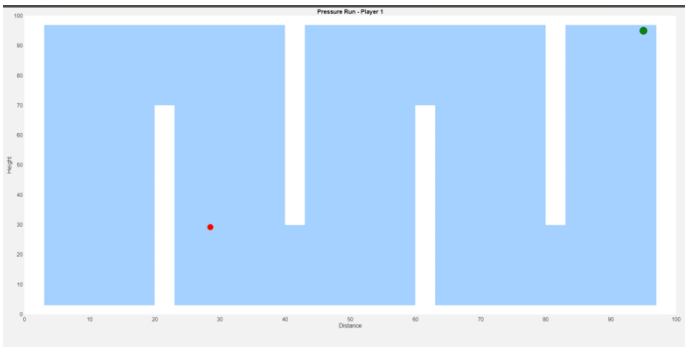
**Results:**





***Figures 6-7:*** Final breadboard setup of the interactive game controller circuit. The blue 3D-printed joystick houses the ADXL337 accelerometer used to control vertical movement via tilt

(Y-axis), while the black 3D-printed button enclosure contains the small FSR used to control forward movement. Both sensors are connected to an AD3 data acquisition device and integrated on a shared breadboard, with the FSR in a voltage divider configuration and the accelerometer filtered using a low-pass RC filter.




***Figures 8:*** Player 1 navigating through the obstacle maze using the FSR and accelerometer-based joystick system. The red dot represents Player 1's current position. The green dot marks the goal.

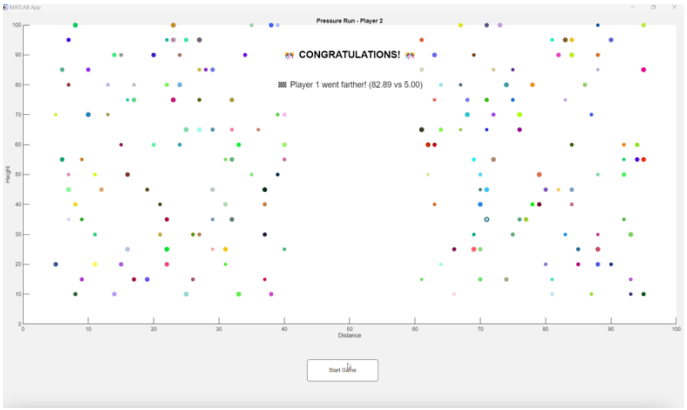



***Figure 9:*** Final game screen displaying the winning result and celebratory confetti animation. Player 1 was declared the winner after reaching a greater distance than Player 2.

**Conclusion (~1/2 page)**

This project successfully brought together real-world sensor data and digital game logic to create a fun, interactive two-player challenge. Throughout the process, we applied key concepts like sensor interfacing, voltage division, signal filtering, collision detection, and live plotting using MATLAB App Designer to build a fully functioning real-time system. Working hands-on with both hardware and software gave us a much deeper understanding of sensors, circuit design, signal conditioning, and real-time programming.

One of the biggest lessons I learned was the importance of not just making everything work but making it reliable and user-friendly too. Using the FSR showed us how delicate some sensors can be, especially when it comes to mounting and mechanical stress, and emphasized the need for stronger and more secure designs when integrating sensitive components into physical systems.

Looking ahead, there are several ways I could improve and expand the project. One idea would be switching the FSR from an analog to a digital setup, which would help make the signal more stable and easier to process. We could also add the X-axis of the accelerometer to the controls, allowing for full two-dimensional movement instead of just vertical adjustments, creating a much more dynamic gameplay experience. Adding more maze levels, obstacles, or bonus pickups could make the game even more challenging and fun for players.

From a hardware standpoint, securing the FSR better would protect it during gameplay to ensure more consistent performance. I would also like to build even more UNC-Chapel Hill spirit into the game, such as including UNC colors, logos, or references to iconic campus landmarks within the design to make it feel even more personal and meaningful to the university.

Overall, this project was an incredibly rewarding experience that pushed me to think creatively while strengthening our skills in hardware integration, circuit design, and real-time programming.