Internet Connected Skee-Ball Machine

Andrew Davis
James Le
Smile Tongkaw

Project Description

The purpose of this machine is to provide entertainment to the targeted audience. The machine allows a player to play a skeeball game to view their final score on the internet. Specifically, the scores were updated to a google spreadsheet where scores were calculated and shown in a cell box. Three sensors were used in this project, each of which was installed near one of the three holes on the machine's backboard. Each of these sensors added 10 points to the user's total score. If the ball passes through the lowest hole, only 10 points will be added to the score since the ball passed only one sensor in the dispenser mechanism. If the ball were launched into the middle hole, it would pass by two sensors on the way down through the dispenser mechanism and add 20 points total. If the ball passes through the third hole, it would pass three sensors and add 30 points to the score. A speaker was also included in the machine, which let the person know when to start/stop throwing the balls down the runway. The speaker would activate for a second at the beginning of the time and activate for a second again 30 seconds after the initial buzz of the speaker. Three LEDs were also utilized, acting as a physical way to let the user know how much time they had left. The LED would initially be lit, but once the user starts the machine, the LED would turn off one by one. The buzzer would then sound, and after that, one of the LEDs would light up after 10 seconds had passed, another would activate after 20 seconds, and the final LED would turn on after 30 seconds. A button was included too to let the user activate the machine. Once the machine had activated the second buzzer, an IFTTT in the Particle attached to the machine would upload the user's score to a google spreadsheet. After the score is uploaded, the user can scan a QR code to access the spreadsheet and put in their name if they would like to.

Code Description

The core of the program contains 4 sections: the initialisations, startup sequence, the running sequence, and the scoring mechanism. The initialisation section instantiates all actuators and determines any timed actions (e.g.: time interval between each data recorded by the ultrasonic sensors, time interval for each LED to turn on as an indicator for the game’s timer, timer for the buzzer).

The startup sequence consists of 3 lit-up LEDs and the running sequence is on standby. When the button is pressed, the running sequence is activated: each of the 3 LEDs will turn off every second as a 3-second countdown before the game begins, which is also indicated by a buzzer going off. The code also utilizes Photon’s built-in timer function. The timer function allows certain codes to execute once the preset timer runs out.

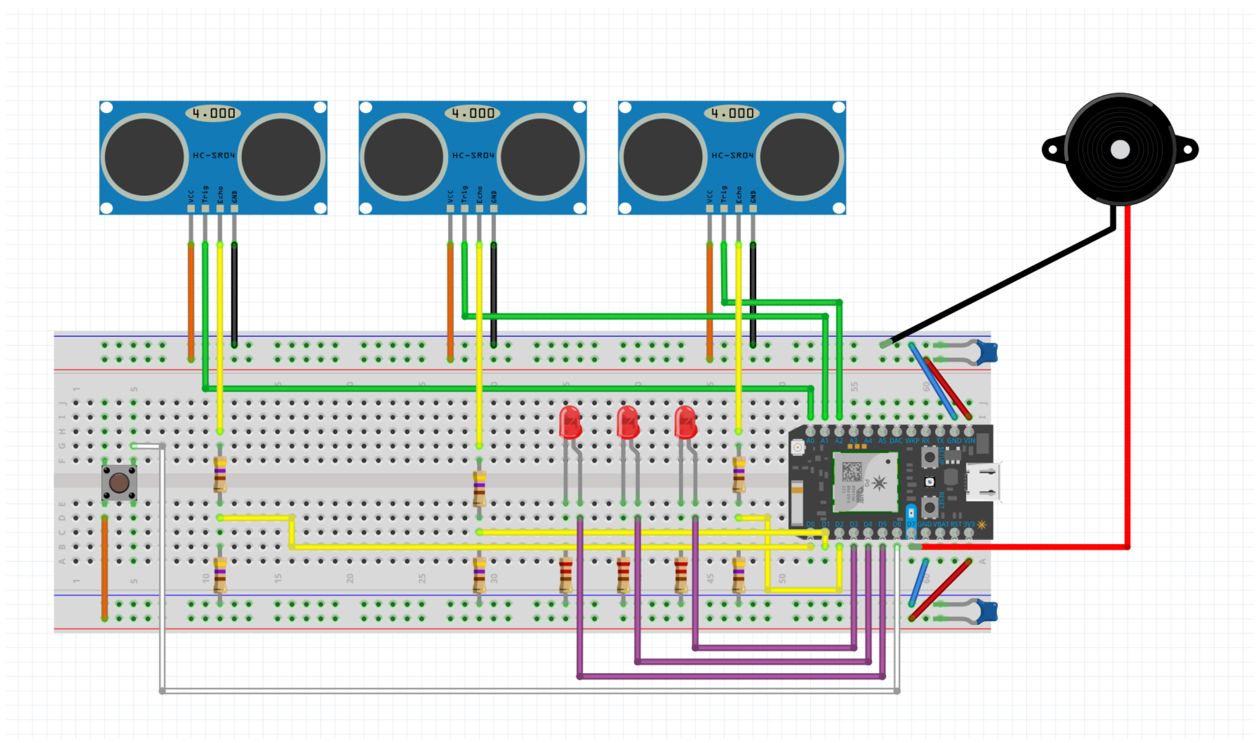
Once the first three seconds of the timer has passed, the running sequence will activate. During the running sequence, the player is allowed to score points until 30 seconds have passed since the buzzer activated. Every 10 seconds, one of the LEDs will turn on which symbolize how much time you have left.

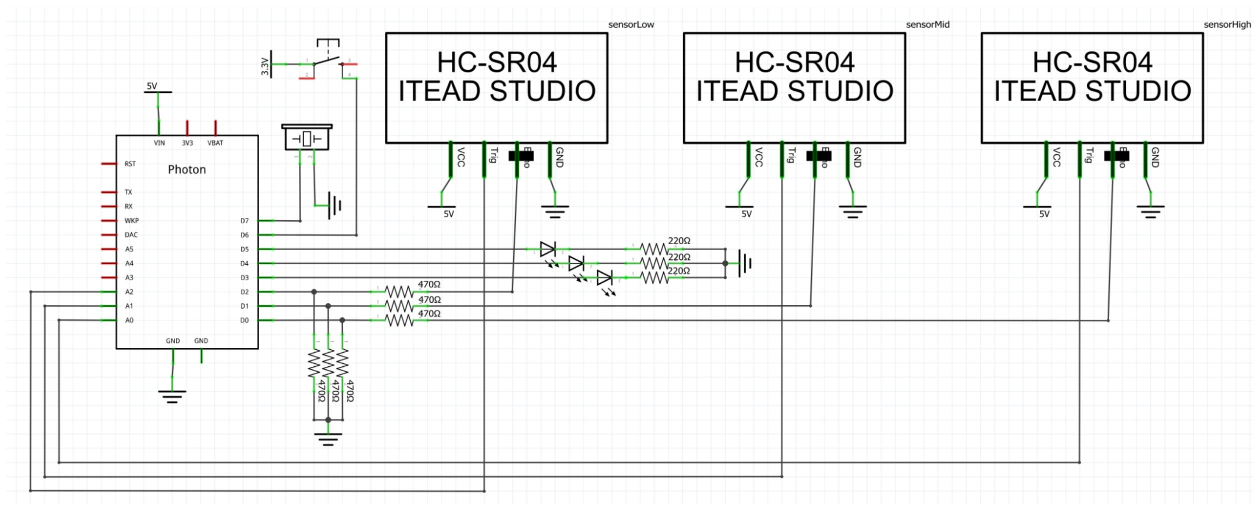
The scoring mechanism works based on the principle that if an object that is in range of the sensor goes out of range, 10 points would be added to the total score for the runtime. The sensors are lined up in such a way such that if a ball is placed in the highest hole, the ball will pass the top, middle, and bottom sensors in the dispenser mechanism and 30 points will be added to the total score. If the ball gets into the middle hole, it will pass the middle and bottom sensors on the way down and thus 20 points will be added to the score. If the ball gets into the lowest hole, the ball will only pass the bottom sensor and 10 points will be added to the score.

Once the running sequence is complete the buzzer will go off again and the 3rd LED will light up, showing the user is out of time. The user is not allowed to score anymore and the total score from that runtime will be uploaded to a google spreadsheet using an IFTTT that is activated when the running sequence is complete. After this, the score is set back to 0 and the entire code can be run again by pressing the button.

The trickiest part of the entire coding process was the process of making all actuators function together as intended. Because this is our first time using HC-SR04 sensors, we had to read documentation and look into different instructions available on the internet. Initially we came across the issue where the sound waves of one sensor would be picked up by other sensors, hence triggering false signals. To counter this problem, we had to make holes on the opposite wall where the sensors are housed to allow sound waves to escape the ball dispenser, padding had been added to the backboard and dispenser in order to reduce sound created by the ball hitting the machine, and we had to rewire the trig output to analog pin and echo output to digital pin (instead of both being wired to digital pins like the beginning). The less tricky part (not to imply that it was any easy) is the scoring mechanism. The sensors continuously emit ultrasonic sound waves to detect objects in their range, so there needed to be an algorithm to appropriately register scores when a ball passes a sensor. Plus, by default, the sensors send out a sound wave every 1 millisecond. This could cause false signals due to environmental factors, so the frequency had to be adjusted to 60 milliseconds.

Electronics Diagram





Code Listing

```
// Date: May 2nd 2022
// Name: Andrew Davis, James Le, Smile Tongkaw
// Course number: EE 1301
// Term: Spring 2022
// Lab/assignment number: IoT Final Project
// Short Program Description: Skeeball Machine



#include "Particle.h"
#include "HC_SR04.h"

HC_SR04 sensorLow = HC_SR04(A2, D2, 7.0, 999);        // (trigPin, echoPin,
minDist, maxDist)
HC_SR04 sensorMid = HC_SR04(A1, D1, 7.0, 999);
HC_SR04 sensorHigh = HC_SR04(A0, D0, 7.0, 999);
double distSensorLow, distSensorMid, distSensorHigh;
double distLastSensorLow, distLastSensorMid, distLastSensorHigh;
const int button = D6;
int buttonLast = LOW;
const int led1 = D3, led2 = D4, led3 = D5;
const int buzzer = D7;
int totalScore = 0;
unsigned long int timeToRecordDist;
unsigned long int timeToBuzz;
Timer led3Timer(10000, led3Run);
Timer led2Timer(20000, led2Run);
Timer led1Timer(30000, led1Run);
Timer buzzerTimerOn(30000, buzzerOn, true);
Timer buzzerTimerOff(31000, buzzerOff);
Timer endTimer(30000, endSequence, true);



void led3Run() {
  digitalWrite(led3, HIGH);
}
void led2Run() {
  digitalWrite(led2, HIGH);
}
void led1Run() {
  digitalWrite(led1, HIGH);
}
void buzzerOn() {
  digitalWrite(buzzer, HIGH);
}
void buzzerOff() {
  digitalWrite(buzzer, LOW);
}
void endSequence() {
  Particle.publish("gameEnd", String(totalScore));
  totalScore = 0;
}



void setup() {
  Serial.begin(9600);
  pinMode(button, INPUT_PULLDOWN);
  pinMode(led1, OUTPUT); pinMode(led2, OUTPUT); pinMode(led3, OUTPUT);
  pinMode(buzzer, OUTPUT);
  timeToRecordDist = millis() + 60;
  timeToBuzz = millis() + 100;
  Particle.variable("totalScore", totalScore);

  // Idle phase: 3 LEDs on, awaiting button push
  digitalWrite(led1, HIGH); digitalWrite(led2, HIGH); digitalWrite(led3,
HIGH);
}


void loop() {
  // Initialisation
  unsigned long int currentTime = millis();
  int buttonNow = digitalRead(button);
  if (currentTime > timeToRecordDist) {                    // Record
distance every 60 ms
    distSensorLow = sensorLow.getDistanceCM();
    distSensorMid = sensorMid.getDistanceCM();
    distSensorHigh = sensorHigh.getDistanceCM();
    timeToRecordDist += 60;
  }


  // Scoring mechanism
  if (distSensorLow != -1) {
    if (distLastSensorLow == -1) {
      totalScore += 10;
    }
  }
  distLastSensorLow = distSensorLow;

  if (distSensorMid != -1) {
    if (distLastSensorMid == -1) {
      totalScore += 10;
    }
  }
  distLastSensorMid = distSensorMid;

  if (distSensorHigh != -1) {
    if (distLastSensorHigh == -1) {
      totalScore += 10;
    }
  }
  distLastSensorHigh = distSensorHigh;




  // Countdown starts upon button push
  if (buttonNow == HIGH && buttonLast == LOW) {
    delay(1000);
    digitalWrite(led3, LOW); delay(1000);
    digitalWrite(led2, LOW); delay(1000);
    digitalWrite(led1, LOW); delay(1000);
    digitalWrite(buzzer, HIGH); delay(1000); digitalWrite(buzzer, LOW);

    // Game starts
    totalScore = 0;
    led3Timer.start();
    led2Timer.start();
    led1Timer.start();
    buzzerTimerOn.start();
    buzzerTimerOff.start();
    endTimer.start();
  }
  buttonLast = buttonNow;


  // Data printed into serial monitor to monitor scores (FOR DEBUGGING
PURPOSE ONLY)
  Serial.print(distSensorLow); Serial.print(", ");
Serial.print(distSensorMid); Serial.print(", ");
Serial.print(distSensorHigh); Serial.print(": ");
Serial.println(totalScore);

}
```