Whack-A-Mole Game Report - Group 11

Code and Hardware: Emirhan Çalışkan, Mehmet Akif Pekşen, Ahmet Halil Yamlı
Design: Ahmet Burak Durukanlıoğlu, Yalçın Eren Onat
Video: Ahmet Burak Durukanoğlu, Ali Eren Kara
Report:Ayşe Naz Özbek
Power point slides: Ali Eren Kara
People in the group that hasn't participated in the Project: Adem Erkiletlioğlu, Ramazan Atakan Giray, Ahmet Nazım Güner

Whack-a-mole is an arcade game where players use a mallet to hit plastic moles that randomly comes out of the holes on a board. The goal is to score as much point as you can in the given time.

In arduino version we made, we made it into a reflex challange which the player has to touch the sensors in the given time when the LED light is lit.
At the beginning of the game, score LEDs light to inform the player that the game has started. Then random leds next to the sensors light randomly, without a specific order. The player has to touch the sensor that has its LED light lit in a randomized time. In this setup, ultrasonic sensors are used to catch movements. If the sensor is touched in time, the point counter increases by one and if not, it stays the same and the LED goes off. Then another random LED light next to a sensor is lit. The game continues like this while the point counter is increasing. At the end of the game, which means 20 times the random LEDs are on and off, the score board LEDs light based on the point counter value at the time. This kind of system can be used to test athletes reflexes etc.

```
const int ledPin1 = 2;  // Pin declarations and conter variable that counts points
const int ledPin2 = 3;
const int ledPin3 = 4;
const int ledPin4 = 5;
int counter = 0;

const int trigPin1 = 6;  // Ultrasonic Sensor trigger and echo pin declarations
const int echoPin1 = 7;

const int trigPin2 = 8;
const int echoPin2 = 9;

const int trigPin3 = 10;
const int echoPin3 = 11;

const int trigPin4 = 12;
const int echoPin4 = 13;
```

In this part, the pins for the LEDs and ultrasonic sensors are defined and connected to the digital pins.

```
void setup() {
  pinMode(ledPin1, OUTPUT); // Led pins set to output mode
  pinMode(ledPin2, OUTPUT);
  pinMode(ledPin3, OUTPUT);
  pinMode(ledPin4, OUTPUT);

  pinMode(trigPin1, OUTPUT);  // Setting the trig and echo pins of the Ultrasonic Sensors for output and input mode
  pinMode(echoPin1, INPUT);

  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2, INPUT);

  pinMode(trigPin3, OUTPUT);
  pinMode(echoPin3, INPUT);

  pinMode(trigPin4, OUTPUT);
  pinMode(echoPin4, INPUT);

  randomSeed(analogRead(0));  // Function to create a random number

  Serial.begin(9600); // Serial Communication starts
}
```

In the setup() function, the pin modes for the LEDs and ultrasonic sensors are set. The LED pins are set as OUTPUT, and the ultrasonic sensor pins are set as either OUTPUT or INPUT. The randomSeed() function is called to initialize the random number generator with a random value from the analog pin 0.

```

float measureDistance(int trigPin, int echoPin) { // Function to measure distance using Ultrasonic Sensors
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  float duration = pulseIn(echoPin, HIGH); // Measuring the duration of the signal from the echo pin
  float distance = (duration * 0.0343) / 2; //Measuring the distance (speed of the sound wave=0.343 m/s)

  return distance;
}
```

The measureDistance() function is used to measure the distance using an ultrasonic sensor. First, the trigger pin is set LOW for 2 microseconds, then set to HIGH for 10 microseconds, and then set back to LOW again. This sequence triggers the ultrasonic sensor to send a wave. The pulseIn() function is then used to measure the time it takes for the ultrasonic wave to come back. The distance is calculated using the formula distance = (duration * 0.0343) / 2 assuming the speed of the sound being 340 m/s

```
61
62  void led1() {
63    digitalWrite(ledPin1, HIGH);
64    delay(600);
65    digitalWrite(ledPin1, LOW);
66    float distance = measureDistance(trigPin1, echoPin1);
67    Serial.print("1 nolu LED yandı, Mesafe: ");
68    Serial.print(distance);
69    if (distance < 17) {
70      Serial.println(" cm - Başarılı");
71      counter++;
72    } else {
73      Serial.println(" cm");
74    }
75  }
76
77   void led2() {
78    digitalWrite(ledPin2, HIGH);
79    delay(600);
80    digitalWrite(ledPin2, LOW);
81    float distance = measureDistance(trigPin2, echoPin2);
82    Serial.print("2 nolu LED yandı, Mesafe: ");
83    Serial.print(distance);
84    if (distance < 17) {
85      Serial.println(" cm - Başarılı");
86      counter++;
87    } else {
88      Serial.println(" cm");
89    }
90   }
91
92   void led3() {
93    digitalWrite(ledPin3, HIGH);
94    delay(600);
95    digitalWrite(ledPin3, LOW);
96    float distance = measureDistance(trigPin3, echoPin3);
97    Serial.print("3 nolu LED yandı, Mesafe: ");
98    Serial.print(distance);
99    if (distance < 17) {
100     Serial.println(" cm - Başarılı");
101     counter++;
102   } else {
103     Serial.println(" cm");
104   }
105  }
106
107 void led4() {
108   digitalWrite(ledPin4, HIGH);
109   delay(600);
110   digitalWrite(ledPin4, LOW);
111   float distance = measureDistance(trigPin4, echoPin4);
112   Serial.print("4 nolu LED yandı, Mesafe: ");
113   Serial.print(distance);
114   if (distance < 17) {
115     Serial.println(" cm - Başarılı");
116     counter++;
117   } else {
118     Serial.println(" cm");
119   }
120 }
```

In this section of the code, these led(n) functions control the LEDs and measure the distance using the datas from the ultrasonic sensor. First, the LED pin is set to HIGH to turn on the LED, then there is a delay of 600 milliseconds, and then the LED pin is set to LOW to turn off the LED. That is to signal the player to touch the sensor. The measureDistance() function is called to measure the distance using the ultrasonic sensor. The distance is then printed to the serial monitor along with a success message if the distance is less than 17 centimeters, which means the player reached the sensor on time, and the counter variable is increased.

```
115
116 void game(){      // Function that runs the game
117
118   for(int i = 1; i < 21; i++){
119     int randomLED = random(1, 5);
120     switch (randomLED) {
121       case 1:
122         led1();
123         break;
124       case 2:
125         led2();
126         break;
127       case 3:
128         led3();
129         break;
130       case 4:
131         led4();
132         break;
133     }
```

The game() function runs the main game loop to keep the game repeating the led functions. It repeats 20 times, randomly selecting an LED to light up using randomLED function with for loop. The led(n)() function is called to control the LED and measure the distance repeatedly.

```
133
134     int led_time = random(20,50); // Setting random delay times between 0.4 - 2.5
135     delay((led_time)*(led_time));
136     }
137 }
```

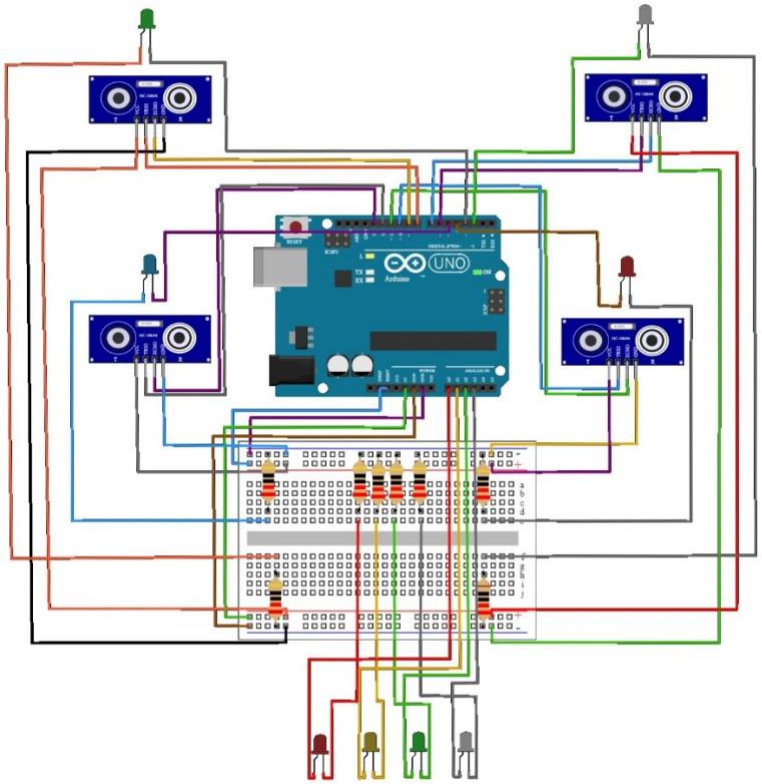
The led_time variable is set to a random value between 0.4 and 2.5 seconds, which determines the the the LED stays on. The program then waits for this duration using the delay() function.

```
138
139   void loop() {
140     digitalWrite(A2,HIGH);  // Turning individual leds on to notify the player
141     delay(1000);
142     digitalWrite(A3,HIGH);
143     delay(1000);
144     digitalWrite(A1,HIGH);
145     delay(1000);
146     digitalWrite(A0,HIGH);
147     delay(2000);
148     digitalWrite(A3,LOW);
149     digitalWrite(A2,LOW);
150     digitalWrite(A1,LOW);
151     digitalWrite(A0,LOW);
```

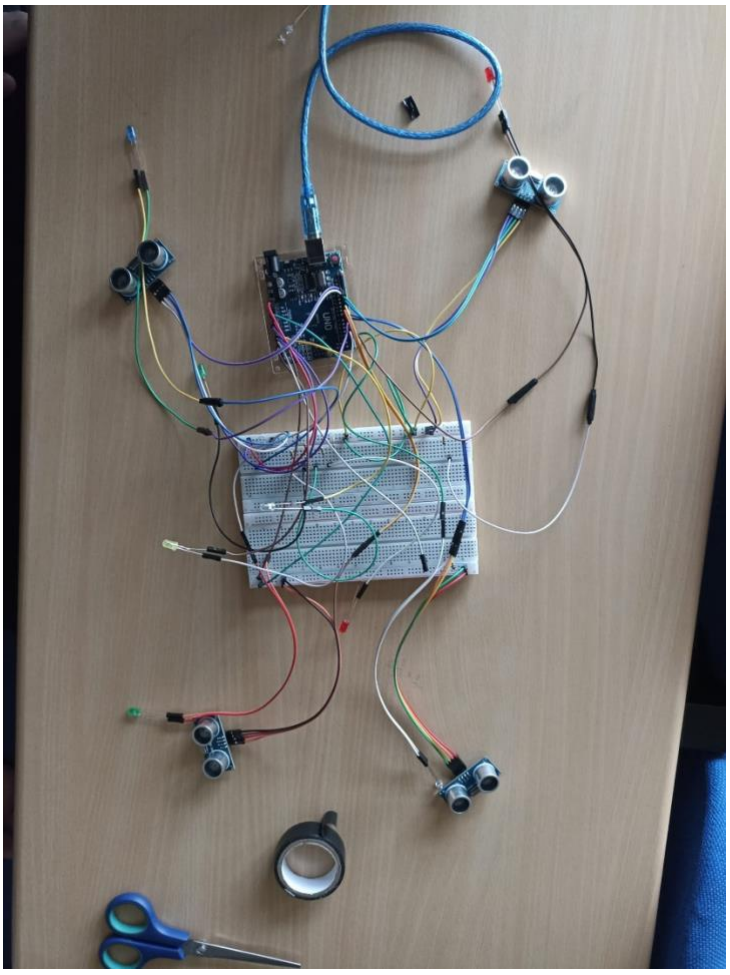
The loop function controls the game flow. This section of the code is to notify the player to start the game. Lights turn on in an order then all goes off.

```
153
154   game(); // Run the game
155   Serial.print(counter); // Show the result/score of the player
156   Serial.println("/20");
157
158   if(counter <= 10){       // Turn on the related central led (Non-mole led) based on the player's score
159   delay(2000);
160   digitalWrite(A2,HIGH);   // 0-10 Red , 11-15 Yellow , 16-19 Green , 20 White
161   }
162   else if(counter <= 15){
163   delay(2000);
164   digitalWrite(A3,HIGH);
165   }
166   else if(counter < 20){
167   delay(2000);
168   digitalWrite(A1,HIGH);
169   }
170   else if(counter == 20){
171   delay(2000);
172   digitalWrite(A0,HIGH);
173   }
174
175   while(1)  // Infinite loop to wait for user action (unplugging or reset button)
176   {}
177 }
```

The game() function is called to run the main game loop in this part of the loop. After the game is finished, the player's score is printed to the serial monitor, each sensor reached on time is equal to one point. Based on the score, different LEDs are turned on to provide feedback to the player. 0-9 points lights red LED, 10-14 points lights yellow LED, 15-19 points light green LED and 20 points lights white LED. In the end, the program runs an infinite loop to prevent it from running any more.



This was the simplified sketch of the arduino hardware of this project. There is 4 sensors and each sensor gets a LED light. And then there is the score board to show the score of the player, which is made with 4 LEDs turning on based on the score.



This system is the hardware inside of the gameboard box that has been made. Its based on the sketch above.

And finally, this the gameboard of the game. The hardware system of the arduino above is inside the box. Each corner has a sensor and a LED belonging to the sensor. In the middle of the board, there is the score board made with 4 LEDs turning on based on the score.

The video of the project:
https://youtu.be/MuRw7xmhXck