YAŞAR UNIVERSITY

COMP 3328: EMBEDDED SYSTEMS PROJECT

PAWPLAY PET ENTERTAINER

INTERACTIVE GADGET FOR AMUSING PETS

BİLGEHAN BAYRAMOĞLU

21070001062

TERM PROJECT ADVISOR: ASSIST.PROF.(PHD) ÖMER ÇETİN

ENGINEERING FACULTY

COMPUTER ENGINEERING PROGRAM

BORNOVA / İZMİR
05-2024

APPROVAL PAGE

I acknowledge and declare that I have personally worked in accordance with academic ethical principles during the preparation of this project. I accept and undertake that if the contrary situation is detected in any way, my project work will not be accepted and will not be evaluated, and I will be subject to sanctions announced by the university administration.

Student: **Signature:**

Bilgehan / Bayramoğlu
21070001062
25/05/2024

ABSTRACT

PAWPLAY PET ENTERTAİNER: AN INTERACTİVE GADGET FOR AMUSİNG PETS

Bayramoğlu, Bilgehan

COMP 3328: Embedded Systems Project Term Project,

Advisor: Assist. Prof.(PhD) Ömer ÇETİN

May 2024

The amusement and engagement of pets through interactive play are essential for their well-being and happiness. This project focuses on the development of "PawPlay Pet Entertainer," an innovative gadget designed to entertain pets using a combination of motion sensors, LED lights, a buzzer, and a servo motor. The gadget aims to create an interactive environment where pets can engage in playful activities triggered by their movements.

Utilizing the Arduino Mega microcontroller, the device coordinates various input and output components to generate dynamic and unpredictable responses, thus keeping pets intrigued and entertained. The system's motion detection capabilities are powered by PIR sensors, which activate visual and auditory stimuli via LED lights and a buzzer. Additionally, the servo motor introduces random movements to further captivate the pet’s attention.

The development and simulation of this project are carried out on the WOKWI online emulator using Arduino C. This environment allows for thorough testing and refinement of the gadget's functionalities before physical implementation. The results from the simulation indicate that the "PawPlay Pet Entertainer" can effectively engage pets, offering a stimulating playtime experience.

Keywords: Arduino Mega, pet entertainment, interactive gadget, PIR motion sensor, servo motor, WOKWI emulator, C, pet well-being.

TABLE OF CONTENTS

1. CHAPTER: PROBLEM DEFINITION

Pets play a significant role in the lives of their owners, contributing to companionship and emotional well-being. However, ensuring that pets remain engaged and entertained, especially in the absence of their owners, can be challenging. Traditional pet toys often fail to sustain a pet's interest over time, leading to boredom and inactivity, which can affect a pet's physical and mental health.

Traditional pet toys often lack the complexity and interactivity needed to maintain a pet's interest over long periods. Simple toys like balls or plush toys might excite a pet initially but soon lose their appeal. Once the novelty wears off, pets can become disinterested, leading to periods of inactivity. This lack of stimulation can contribute to various health issues, including weight gain, muscle atrophy, and behavioral problems such as excessive barking, chewing, or scratching.

Interactive pet toys and gadgets offer a promising solution by providing continuous and varied stimulation. These devices can respond to a pet's actions, creating a more engaging and enriching environment. By incorporating elements that react to movement, sound, and other stimuli, interactive toys can mimic the unpredictable nature of real-life play, keeping pets mentally and physically active.

2. CHAPTER: THE GOAL OF THE PROJECT

The goal of the PawPlay Pet Entertainer project is to develop an innovative and interactive gadget designed to engage pets in playful activities, providing entertainment and stimulation to improve their overall well-being. This project focuses on creating a system that can autonomously respond to pet movements and behaviors with various stimuli, ensuring that pets remain active and entertained even when their owners are not present.

The PawPlay Pet Entertainer is intended for use in domestic environments where pets, particularly cats and dogs, require mental and physical stimulation. The device can be used in various locations within the home, such as living rooms, pet play areas, and even outdoors, provided there is access to a power source. The primary usage area is aimed at pet owners who seek to enhance their pets' quality of life through interactive play.

The PawPlay Pet Entertainer will incorporate several advanced features designed to captivate and amuse pets:

- **Motion Detection:** Equipped with PIR motion sensors, the gadget will detect the movement of pets and trigger interactive responses.
- **Visual Stimulation:** LED lights will be used to create engaging visual effects that attract pets.
- **Auditory Stimulation:** A buzzer will emit sounds to further engage pets and encourage playful behavior.
- **Unpredictable Movements:** A servo motor will create random and unpredictable movements, adding an element of surprise and keeping pets intrigued.

3. CHAPTER: HARDWARE

3.1. Microprocessor / Microcontroller

- The microcontroller chosen for the PawPlay Pet Entertainer project is the Arduino Mega.
- Operating at 16MHz, the Arduino Mega 2560 delivers reliable performance, ensuring smooth execution of tasks and responsiveness to input.
- Featuring 4 serial ports, the microcontroller enables communication with external devices and facilitates data exchange, enhancing the project's versatility.

3.2. Diagram Defination of Project

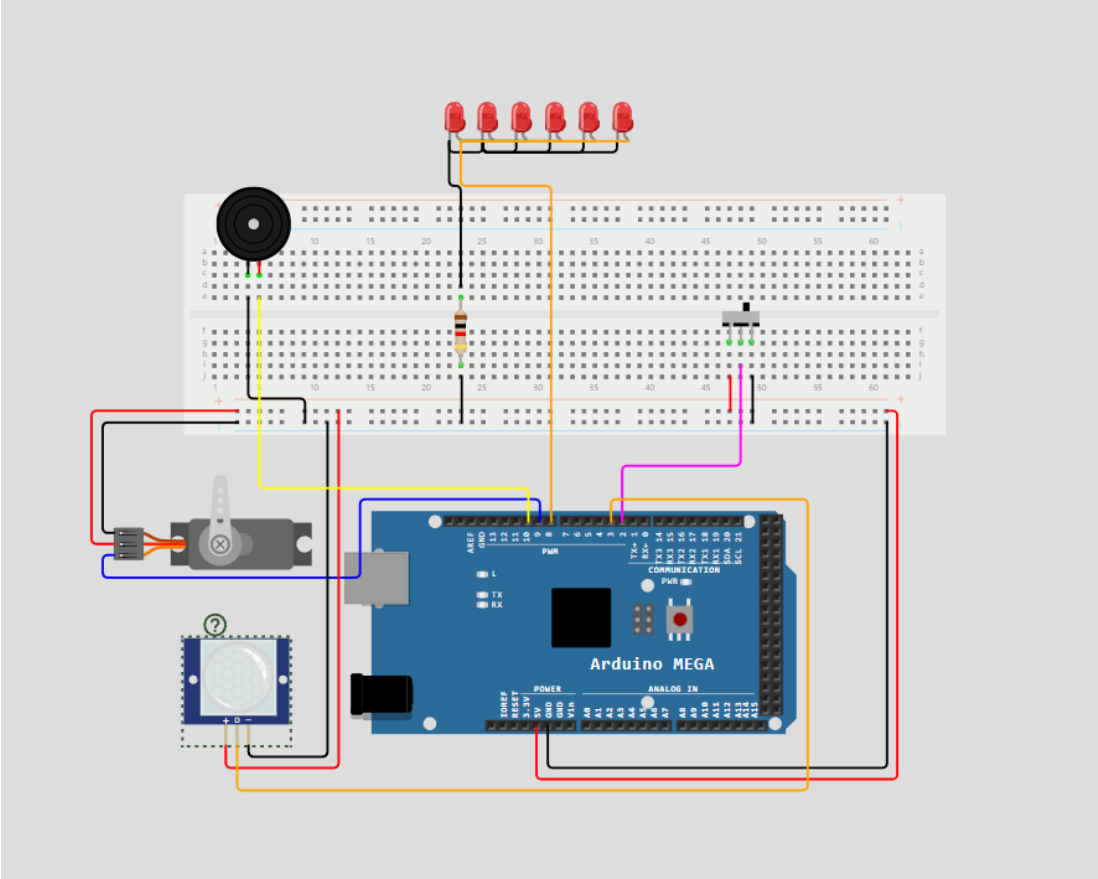


Figure 2.1. Image Description

This pet interactive gadget features LED lights for visual attraction and a slide switch for easy on/off control. A PIR motion sensor detects pet movement, triggering engaging actions. The gadget includes a buzzer to emit playful sounds and a servo to create unpredictable movements, enhancing surprise and amusement for pets.

3.3. Other Input / Output Devices Used within the Scope of the Project

Type	I/O Device	Purpose of usage
Display	*LED*	Add visual effects and attract pets to interact with the gadget
Input	*Slide Switch*	By incorporating a slide switch into your gadget design, users can easily turn the toy on or off as desired, providing them with convenient control over its operation
Sensor	*PIR Motion*	Detect pet movement to trigger actions and interactions.
Output	*Buzzer*	Emit sound to engage pets in playful activities.
Output	*Servo*	Utilized to create unpredictable movements, enhancing the element of surprise and amusement for pets.

3.4. Cost of Project

Arduino Mega 2560

Cost: 502.00 TL

Source: Teknoartshop

Description: Main microcontroller for managing all sensors and actuators.

PIR Motion Sensor

Cost: 97.00 TL

Source: Rakun Robotik

Description: Detects pet movement to trigger actions and interactions.

Servo Motor (MG995)

Cost: 300.00 TL

Source: ikizsoft

Description: Creates unpredictable movements to engage pets.

LED (Pack of 10)

Cost: 53.00TL (pack of 10)

Source: Argema

Description: Provides visual effects to attract pets.

Slide Switch

Cost: 26.00 TL

Source: Sonexson

Description: Allows users to easily turn the gadget on or off.

Buzzer

Cost: 13.34 TL

Source: Robotistan

Description: Emits sound to engage pets in playful activities.

Resistors

Cost: 1.96 TL (pack of 10)

Source: robotistan

Description: Required for various circuit connections.

Breadboard

Cost: 221.00TL

Source: direnc.net

Description: Used for prototyping and testing the circuit connections.

Jumper Wires (Pack of 40)

Cost: 26.05 TL

Source: ROBO90

Description: For making connections between components.

Total Cost: 1240,00 TL

4. CHAPTER: SOFTWARE DEVELOPMENT

The software development for the PawPlay Pet Entertainer project involves designing and implementing the code that will control the various hardware components, including the microcontroller, sensors, LED, servo motor, and buzzer. The software is developed using the Arduino IDE and tested on the WOKWI emulator to ensure functionality before deployment to the physical hardware.

4.1. Functions

The software for the PawPlay Pet Entertainer is structured into several key functions that manage the interactions between the different components. These functions are designed to handle the input from the slide switch and the PIR motion sensor, as well as to control the outputs to the LED, servo motor, and buzzer.

- Random Servo Movement Function: Generates random angles for the servo motor to move, creating unpredictable movements that engage the pet, and engages the pet.
- LED Blink Function: Controls the blinking pattern of the LED when motion is detected, and engages the pet.
- Buzzer Control Function: Activates the buzzer to emit sounds when motion is detected, and engages the pet.

4.2.Constants & Variables

The software uses several constants and variables to manage the state of the hardware components and store intermediate values during execution. These constants and variables are defined at the beginning of the code.

5. CHAPTER: CODE

5.2. Arduino Code

```
#include <Servo.h>
#include <util/delay.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "wiring_private.h"
#include "pins_arduino.h"

#define SWITCH_PIN 2
#define LED_PIN 8
#define BUZZER_PIN 10
#define SERVO_PIN 9
#define PIR_PIN 3
#define bit2(b) (1UL << (b))

Servo servoMotor;

void delay2(unsigned long ms) {
    while (ms > 0) {
        _delay_ms(1);
        ms--;
    }
}

int generateRandom(int minimum, int maximum) {
    return minimum + rand() % (maximum - minimum + 1);
}

void digitalWrite2(uint8_t pin, uint8_t val)
{
    uint8_t timer = digitalPinToTimer(pin);
    uint8_t bit2 = digitalPinToBitMask(pin);
    uint8_t port = digitalPinToPort(pin);
    volatile uint8_t *out;

    if (port == NOT_A_PIN) return;

    out = portOutputRegister(port);
```

```
44.
45.     uint8_t oldSREG = SREG;
46.     cli();
47.
48.     if (val == LOW) {
49.         *out &= ~bit2;
50.     } else {
51.         *out |= bit2;
52.     }
53.
54.     SREG = oldSREG;
55. }
56.
57. int digitalRead2(uint8_t pin)
58. {
59.     uint8_t bit2 = digitalPinToBitMask(pin);
60.     uint8_t port = digitalPinToPort(pin);
61.
62.     if (port == NOT_A_PIN) return LOW;
63.
64.     if (*portInputRegister(port) & bit2) return HIGH;
65.     return LOW;
66. }
67.
68. void pinMode2(uint8_t pin, uint8_t mode)
69. {
70.         uint8_t bit2 = digitalPinToBitMask(pin);
71.         uint8_t port = digitalPinToPort(pin);
72.         volatile uint8_t *reg, *out;
73.
74.         if (port == NOT_A_PIN) return;
75.
76.         reg = portModeRegister(port);
77.         out = portOutputRegister(port);
78.
79.         if (mode == INPUT) {
80.                 uint8_t oldSREG = SREG;
81.                 cli();
82.                 *reg &= ~bit2;
83.                 *out &= ~bit2;
84.                 SREG = oldSREG;
85.         } else if (mode == INPUT_PULLUP) {
86.                 uint8_t oldSREG = SREG;
87.                 cli();
88.                 *reg &= ~bit2;
89.                 *out |= bit2;
90.                 SREG = oldSREG;
91.         }
```

```
92. else {
93.                 uint8_t oldSREG = SREG;
94.                 cli();
95.                 *reg |= bit2;
96.                 SREG = oldSREG;
97.         }
98. }
99.
100.  void setup() {
101.      pinMode2(SWITCH_PIN, INPUT_PULLUP);
102.      pinMode2(LED_PIN, OUTPUT);
103.      pinMode2(BUZZER_PIN, OUTPUT);
104.      pinMode2(PIR_PIN, INPUT);
105.      servoMotor.attach(SERVO_PIN);
106.      srand(time(0));
107.  }
108.
109.  void loop()
110.  {
111.      int switchState = digitalRead2(SWITCH_PIN);
112.      int motionState = digitalRead2(PIR_PIN);
113.
114.      if (switchState == LOW && motionState == HIGH)
115.      {
116.          for (int i = 0; i < 3; i++)
117.          {
118.              digitalWrite2(LED_PIN, HIGH);
119.              digitalWrite2(BUZZER_PIN, HIGH);
120.              delay2(100);
121.              digitalWrite2(LED_PIN, LOW);
122.              digitalWrite2(BUZZER_PIN, LOW);
123.              delay2(100);
124.          }
125.
126.          int randomAngle = generateRandom(0, 180);
127.          servoMotor.write(randomAngle);
128.          delay2(250);
129.      }
130.      else {
131.          digitalWrite2(LED_PIN, LOW);
132.          digitalWrite2(BUZZER_PIN, LOW);
133.          servoMotor.write(0);
134.      }
135.  }
136.
137.
```

137.2. WOKWI diagram.json File

```
{
  "version": 1,
  "author": "Anonymous maker",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-breadboard", "id": "bb1", "top": -137.4, "left": 108.4, "attrs": {} },
    { "type": "wokwi-arduino-mega", "id": "mega", "top": 135, "left": 246, "attrs": {} },
    { "type": "wokwi-slide-switch", "id": "sw1", "top": -43.6, "left": 569.5, "attrs": {} },
    {
      "type": "wokwi-led",
      "id": "led1",
      "top": -224.4,
      "left": 320.6,
      "attrs": { "color": "red" }
    },
    {
      "type": "wokwi-resistor",
      "id": "r1",
      "top": -24,
      "left": 316.25,
      "rotate": 90,
      "attrs": { "value": "1000" }
    },
    { "type": "wokwi-servo", "id": "servo1", "top": 103.6, "left": 48, "attrs": {} },
    { "type": "wokwi-pir-motion-sensor", "id": "pir1", "top": 244, "left": 107.82, "attrs": {} },
    {
      "type": "wokwi-buzzer",
      "id": "bz1",
      "top": -151.2,
      "left": 136.2,
      "attrs": { "volume": "0.1" }
    },
    {
      "type": "wokwi-led",
      "id": "led2",
      "top": -224.4,
      "left": 349.4,
      "attrs": { "color": "red" }
    },
    {
      "type": "wokwi-led",
```

```
        "id": "led3",
        "top": -224.4,
        "left": 378.2,
        "attrs": { "color": "red" }
      },
      { "type": "wokwi-led", "id": "led4", "top": -224.4, "left": 407, "attrs": { "color": "red" } },
      {
        "type": "wokwi-led",
        "id": "led5",
        "top": -224.4,
        "left": 435.8,
        "attrs": { "color": "red" }
      },
      {
        "type": "wokwi-led",
        "id": "led6",
        "top": -224.4,
        "left": 464.6,
        "attrs": { "color": "red" }
      }
    ],
    "connections": [
      [ "mega:GND.2", "bb1:bn.50", "black", [ "v35.7", "h366.95" ] ],
      [ "mega:2", "bb1:48b.i", "magenta", [ "v-48", "h188" ] ],
      [ "bb1:bn.40", "bb1:49b.j", "black", [ "v0" ] ],
      [ "bb1:bp.38", "bb1:47b.j", "red", [ "v0" ] ],
      [ "bb1:bn.18", "bb1:23b.j", "black", [ "v0" ] ],
      [ "servo1:GND", "bb1:bn.1", "black", [ "h-9.6", "v-86.4" ] ],
      [ "servo1:V+", "bb1:bp.1", "red", [ "h-19.2", "v-105.5" ] ],
      [ "servo1:PWM", "mega:9", "blue", [ "h-9.6", "v19.4", "h220.8", "v-67.2", "h192" ] ],
      [ "mega:5V", "bb1:bp.50", "red", [ "v45.3", "h309.5", "v-316.8" ] ],
      [ "pir1:GND", "bb1:bn.8", "black", [ "v9.6", "h57.34" ] ],
      [ "pir1:VCC", "bb1:bp.9", "red", [ "v19.2", "h86.4" ] ],
      [ "bb1:bn.6", "bb1:4t.e", "black", [ "v-20.5", "h-20" ] ],
      [ "bb1:5t.e", "mega:10", "yellow", [ "v163.2", "h230.7" ] ],
      [ "pir1:OUT", "mega:3", "orange", [ "v38.4", "h489.46", "v-249.6", "h-172.8" ] ],
      [ "led1:C", "bb1:23t.d", "black", [ "v38.4", "h10" ] ],
      [ "led2:A", "led1:A", "blue", [ "v0" ] ],
      [ "led2:C", "led1:C", "black", [ "v9.6", "h0.4" ] ],
      [ "led1:A", "mega:8", "orange", [ "v38.4", "h76.8" ] ],
      [ "led3:C", "led2:C", "black", [ "v9.6", "h-28.4" ] ],
      [ "led4:C", "led2:C", "black", [ "v9.6", "h-57.2" ] ],
      [ "led5:C", "led2:C", "black", [ "v9.6", "h-86" ] ],
```

```
      [ "led6:C", "led2:C", "black", [ "v9.6", "h-114.8" ] ],
      [ "led3:A", "led1:A", "green", [ "v0" ] ],
      [ "led4:A", "led1:A", "green", [ "v0" ] ],
      [ "led6:A", "led1:A", "orange", [ "v0" ] ],
      [ "led5:A", "led1:A", "orange", [ "v0" ] ],
      [ "sw1:1", "bb1:47b.g", "", [ "$bb" ] ],
      [ "sw1:2", "bb1:48b.g", "", [ "$bb" ] ],
      [ "sw1:3", "bb1:49b.g", "", [ "$bb" ] ],
      [ "r1:1", "bb1:23t.e", "", [ "$bb" ] ],
      [ "r1:2", "bb1:23b.i", "", [ "$bb" ] ],
      [ "bz1:1", "bb1:4t.c", "", [ "$bb" ] ],
      [ "bz1:2", "bb1:5t.c", "", [ "$bb" ] ]
    ],
    "dependencies": {}
  }
```

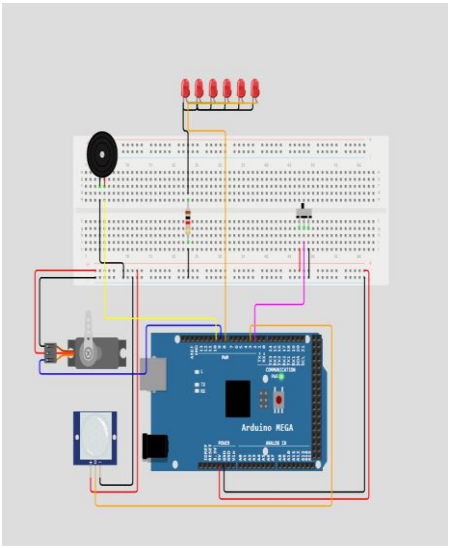
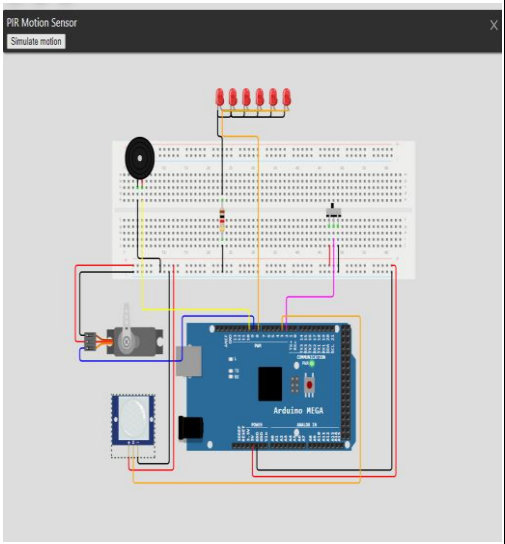
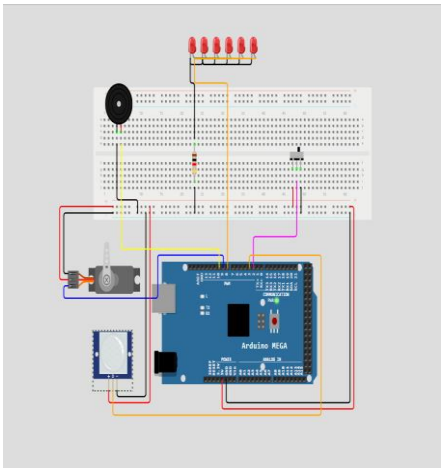
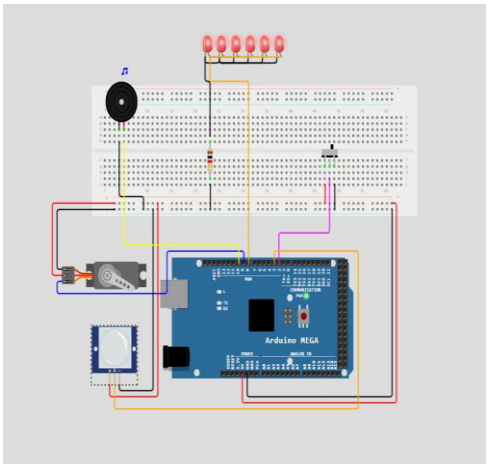
236.2. WOKWI libraries.txt File

```
# Wokwi Library List

# See https://docs.wokwi.com/guides/libraries



# Automatically added based on includes:

Servo
```

236.3. Other WOKWI Files

-

6. CHAPTER: PROJECT RUNTIME SCREENSHOT IMAGES

	
Slide switch is in the off position. The system is on off state and it is not controlling any outputs.	There is a motion but there is no outputs, because of the state of slide switch
	
Slide switch is in the on position. The system is on on state and waiting for input (motion).	system in its ON state with motion detected. In this state, the slide switch is set to the "on" position, activating the system to monitor for motion and control the outputs accordingly.

7. CHAPTER: CONCLUSIONS AND FUTURE WORK

In conclusion, the development and implementation of the PawPlay Pet Entertainer project have demonstrated the feasibility and effectiveness of using an Arduino-based system to create an engaging and interactive toy for pets. The system leverages various sensors and actuators to entertain pets through dynamic and unpredictable interactions, thus addressing the challenge of keeping pets entertained, especially in the absence of their owners.

The project successfully integrated multiple hardware components, including a slide switch, PIR motion sensor, LED, servo motor, and buzzer, all controlled by an Arduino Mega 2560 microcontroller. Key achievements of the project include.

- **Effective Motion Detection:** The PIR motion sensor reliably detects pet movement, triggering the system to engage the pet.
- **Interactive Responses:** The servo motor and LED provide dynamic visual and mechanical stimuli, while the buzzer adds an auditory element to the interaction.
- **User Control:** The slide switch allows users to easily turn the system on and off, providing convenient control over the gadget's operation.

The Arduino code developed for this project effectively manages the interaction between the sensors and actuators. When the system is turned on and motion is detected, the servo motor moves to random positions, the LED blinks, and the buzzer emits sounds, creating an engaging experience for pets. This interactive behavior was successfully simulated in the WOKWI emulator, validating the functionality of the system.

8. future Work

While the current implementation of the PawPlay Pet Entertainer is effective, there are several areas for future improvement and expansion:

1. **Enhanced Interactivity:** Incorporating additional sensors and actuators, such as cameras or more advanced motion detectors, could further enhance the interactivity of the gadget.
2. **Remote Control:** Adding Wi-Fi or Bluetooth connectivity would allow owners to control the gadget remotely via a smartphone app, providing greater flexibility.
3. **Durability and Safety:** Future iterations could focus on making the gadget more durable and safe for pets, using pet-friendly materials and ensuring robust construction.

In summary, the PawPlay Pet Entertainer project demonstrates the potential for creating a sophisticated and engaging toy for pets using embedded systems. By continuing to innovate and expand on this foundation, future developments can provide even more benefits for pets and their owners, making pet ownership a more enjoyable and fulfilling experience.

9. REFERENCES

https://docs.arduino.cc/programming/

https://docs.wokwi.com/