2024

The Bat Box Final Report

Nasir Mendez, Reece Clem, Marquise Allsup
Frostburg State University
12/9/2024

Table of Contents

Introduction

This project explores the use of the Raspberry Pi Zero 2 W and a US100 Ultrasonic Sensor to create a functional "Bat Box," a system designed for measuring distance and temperature. Central to the project are several components:

- **Raspberry Pi Zero 2 W**: A compact, wireless-enabled computer running Raspberry Pi OS Lite.
- **US100 Ultrasonic Sensor**: A device operating in UART mode to measure distance and temperature.
- **T-Cobbler and Breadboard**: Hardware for connecting components.

Our goal was to integrate these technologies, write functional code using the **Thonny IDE**, and develop practical applications for our system. The project highlights challenges and solutions while providing a roadmap for others interested in similar ventures.

Overview of Raspberry Pi Zero 2 W

The Raspberry Pi Zero 2 W, launched in 2021, features:

- A **quad-core ARM Cortex-A53 CPU** clocked at 1GHz.
- 512MB SDRAM for modest computational needs.
- Built-in **Wi-Fi** (2.4 GHz) and **Bluetooth 4.2**.

It supports diverse applications, from robotics to IoT projects. The lightweight Raspberry Pi OS Lite was selected for its memory efficiency and compatibility. Installation involved flashing the OS onto a microSD card using Pi Imager, followed by basic configuration through the command line. Despite its limited memory, the Pi Zero 2 W’s processing power and connectivity made it ideal for our "Bat Box" project.

US100 Ultrasonic Sensor and Integration

The US100 Ultrasonic Sensor calculates distances using the formula:

Distance=Time×Speed of Sound (343 m/s)\text{Distance} = \text{Time} \times \text{Speed of Sound (343 m/s)} Distance=Time×Speed of Sound (343 m/s)

The sensor transmits ultrasonic waves and records the time taken for the echo to return. In UART mode, the sensor communicates with the Raspberry Pi via serial data exchange. Key integration steps included:

1. **Physical Setup**: Connecting components on a breadboard using the T-cobbler.
2. **Software Configuration**: Installing necessary UART packages and configuring serial input via the terminal.
3. **Code Development**: Writing Python scripts in **Thonny IDE** to interpret sensor data and generate outputs.

Challenges included adapting Arduino-based code for UART and ensuring accurate pinout connections. The result was a reliable system capable of delivering accurate distance and temperature measurements.

The Bat Box: Concept and Design

The "Bat Box" serves as a protective shell for the sensor and Raspberry Pi, designed in the shape of a bat to reflect its ultrasonic theme. Its design ensures:

- **Durability**: Protects wiring and hardware from environmental damage.
- **Portability**: Lightweight and compact for deployment in various settings.
- **Functionality**: Outputs data to LEDs, speakers, or LCDs for real-time distance and temperature readings.

This project concept aligns with themes of biomimicry, drawing inspiration from bats' echolocation abilities.

Website Integration

Our project website serves as an information hub, showcasing the Raspberry Pi Zero 2 W and Bat Box. Key features include:

- **Detailed Tutorials**: Step-by-step guides for hardware assembly, OS installation, and software configuration.

- **Project Documentation**: A blog section chronicling our development process and troubleshooting experiences.
- **Interactive Features**: Live demonstrations of sensor data and a Q&A forum for user engagement.

The website bridges the gap between technical details and user accessibility, promoting education and collaboration.

Challenges Faced and Solutions

Key challenges included:

1. **Code Adaptation**: Most sensor scripts were Arduino-specific and not optimized for UART. We rewrote scripts using Python libraries like serial to work with the Raspberry Pi.
2. **Pinout Complexity**: Misconnections risked damaging the board. Careful referencing of GPIO pinout diagrams and testing mitigated this issue.
3. **Initial OS Setup**: Configuring the Pi for serial communication required installing and testing UART packages.

Each challenge was addressed systematically, showcasing problem-solving skills and reinforcing the project's educational value.

Operating Systems for Raspberry Pi

A comparative analysis of operating systems highlighted:

- **Ubuntu**: Robust but resource-intensive.
- **DietPi**: Lightweight but less supported for beginners.

- **Raspberry Pi OS**: Stable but not optimized for low-memory environments.
- **Raspberry Pi OS Lite**: Lightweight, efficient, and well-documented.

We selected **Raspberry Pi OS Lite** for its balance between functionality and efficiency.

Literature Review

Studies on the Raspberry Pi Zero 2 W emphasize its affordability, improved processing power, and diverse applications. Its **quad-core processor** enhances multitasking capabilities compared to earlier models. Limitations include constrained memory, addressed in this project through lightweight OS selection.

Research into ultrasonic sensors corroborates their reliability for proximity detection, with applications in robotics, automotive, and IoT systems. Combining these technologies broadens the scope of innovation.

The Raspberry Pi Zero 2 W has seen significant advancements since its release. Studies have demonstrated its effectiveness in areas such as:

- **IoT Systems**: Researchers have used the Zero 2 W in smart home systems to integrate sensors and actuators, citing its wireless capabilities as a major advantage.
- **Educational Platforms**: Its affordability and simplicity make it a popular tool for teaching programming and electronics. Educators have reported increased student engagement through hands-on projects involving the device.
- **Edge Computing**: The quad-core processor enables the Zero 2 W to handle lightweight machine learning tasks, making it suitable for edge-based data processing.

Studies on integrating the Raspberry Pi Zero 2 W with sensors identify challenges that align with our experiences:

UART Communication: Proper configuration of UART for serial data exchange is a common hurdle. Research emphasizes the need for robust error-handling in code to address data transmission delays or loss.

Memory Constraints: Many projects face difficulties running resource-heavy applications on devices with limited RAM. Solutions often involve optimizing code and selecting lightweight operating systems, as we did with Raspberry Pi OS Lite.

Power Management: While the Zero 2 W is energy-efficient, integrating multiple components can strain power supply systems. Studies recommend using external power sources or capacitors to stabilize power delivery.

Ultrasonic Sensors in Proximity Detection

Ultrasonic sensors like the US100 have been widely studied for their accuracy and reliability in measuring distance. Key applications include:

- **Automotive Systems**: Ultrasonic sensors are integral to parking assistance and collision detection systems, where precision is critical.
- **Industrial Automation**: Factories use ultrasonic sensors for object detection and conveyor system monitoring.

- **Wildlife Monitoring**: Similar to our project, ultrasonic sensors have been employed in ecological studies to track animals and monitor habitats.

These sensors are particularly valued for their resilience in various environmental conditions, including low light and dust, which can hinder optical sensors.

Conclusion

The literature underscores the versatility and growing relevance of the Raspberry Pi Zero 2 W and ultrasonic sensors. By leveraging their strengths and addressing documented challenges, our "Bat Box" project contributes to this expanding field, offering a practical and educational use case. Future work could draw from the latest trends, further enhancing the system's capabilities.