Computer Architecture - University of Basel
Final Project Report
Whack a Mole with Arduino

Yaowen Rui, Lisette Maureira Dominguez, Hana Sorani

February 7, 2025

1 Abstract

Whack-a-Mole is a classic arcade game that originated in 70's Japan. This project recreates it using Arduino, with LED buttons representing the moles appearing to be whacked. It enhances the traditional gameplay by adding new challenges, such as accelerating the moles' appearances, interactive sound effects, and visual display of game states. The primary technical challenge involves properly wiring the connections and integrating all necessary hardware components. For detailed achievements, refer to section 3.1: Game Logic.

1.1 Project Overview

The project is divided into two main parts: circuit simulation and code testing using Tinkercad, and the pre-processing and assembly of the mechanical and electronic components.

2 Features

2.1 Components

Five LED Buttons: Representing five moles, each button lights up to indicate a "mole" appearance.

Random Mole Selection: A "mole" appears as a randomly selected button with its LED light turned on.

Scoring System: A LCD-screen displays the player's score by tracking successful hits ("whacked" moles) and missed moles (that cause a game over if the counter goes beyond the 3 permissible misses). Additionally, the highest score is stored permanently and displayed every time before new game starts.

Speaker for Sound Effects: Provides audio feedback for actions such as whacking a mole, missing a mole, achieving the highest score, and starting/ending the game.

Hardware requirements:

```
-Arduino Uno              -220 & 1k Ohm Resistors
-Cables                   -Motherboard
-LCD I2C                  -LED push button switch, 60mm, 12V
-4 Ohm 3 Watt Speaker     -Tip120 Transistor
```

2.2 Simulation

The speaker has not been added to the simulation.

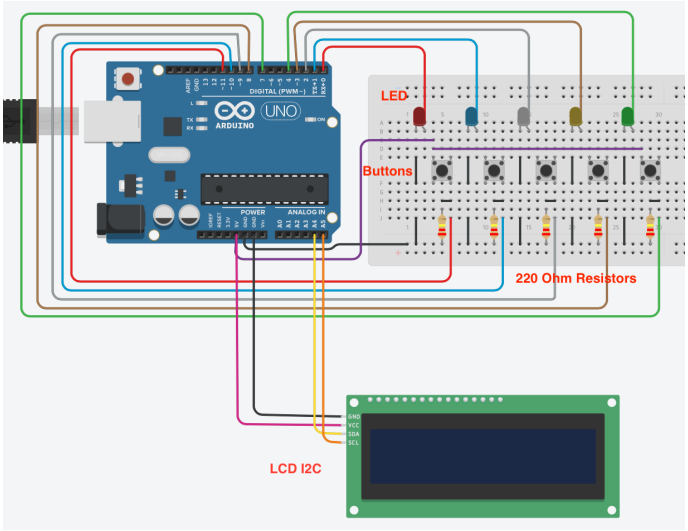

Figure 1: Simulation with Tinkercad

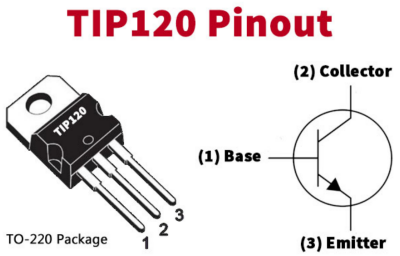

Figure 2: Transistor pins (source)

The speaker was attached to the Arduino through a transistor, with the base pin of the transistor connecting to pin 12 of the Arduino through a 1k Ohm resistor, the negative speaker cable on the transistor's collector pin in the middle, and the positive cable of the speaker to the 5V pin on the Arduino. The transistor's Emitter pin is connected to the negative current of the motherboard.

2.3 Demo

Link to video Demo: YouTube or Google Drive

3 Implementation

3.1 Game Logic

After turning on the power switch, the game begins with the buttons twinkling in sequence to indicate the start. After this, one of the eight buttons will light up at random. The player must press the lit button before the next one lights up. As the game progresses, the moles appear more frequently, increasing the challenge up to a set limit. The speed level change is triggered by the amount of moles that have been whacked by the player (Switch at 10, 25, 35 whacked moles, making up for a

total of 4 levels: Beginner, Amateur, Expert, Legendary). Players are allowed up to three missed moles. Sound effect is triggered by game start/end and whack/miss the moles and reaching the highest score. A new game automatically begins if the player leaves the power switch on.

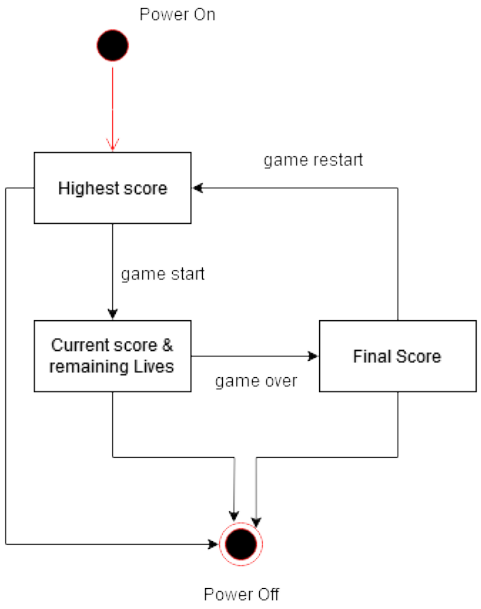

Figure 3: State diagram for the LCD display

3.2 Labor division

Group member	Responsibilities
Hana Sorani	**Hardware:** - Testing individual hardware components (Arduino, buttons, display,...) - Assembling & designing final device **Administration:** - Acquiring any additional material needed for the project
Yaowen Rui	**Hardware:** - Design circuits with Tinkercad & physical assembling of circuits **Code:** - Game logic - Highscore table storing - Debugging - LED sequences Start & Game Over **Administration:** - Managing orders of hardware components - Managing project files (code, slidesets) - Group coordination (communication, setting up meetings)
Lisette Maureira	**Hardware:** - Physical assembling of circuits (speaker) - Testing possible game states for debugging **Code:** - Sound: Composing & coding jingles - Display: Start Animation **Administration:** - Project plan (game elements, goals, milestones...)

3.3 Methodology

LED-Button Pre-Processing: The LEDs on the buttons are designed for 12-volt compatibility, but since all electronics in this project will be powered by an Arduino Nano, the LEDs need to operate at 5 volts. To achieve this, the original resistors in the LED circuit must be replaced with lower 220-ohm resistors.
Button-Press Debouncing: Sometimes, the number of "whacked" moles increases automatically, even when the button is not pressed. This issue occurs because button presses can generate multiple rapid signals (bounces), leading to incorrect or missed hit recognition. To solve this, software debouncing has been implemented to ensure that only a single signal is registered per button press.

3.4 Code overview

Used libraries: LCD, EEPROM

Function	Responsibility
setup	- Initializes hardware components like LEDs, buttons, speaker, and LCD. - Displays the welcome screen and loads the highest score from EEPROM. - Plays the start melody and performs the initial LED sequence.
loop	- Continuously selects a random LED to light up and waits for the corresponding button press. - Tracks successful hits, updates the score, and adjusts the LED speed. - Plays sound effects for successful hits or missed moles. - Checks if the game should end and handles the game-over logic.
isButtonPressed	- Implements software debouncing to ensure only one signal is registered per button press.
loadHighestScore	- Reads the highest score from EEPROM
saveHighestScore	- Updates and saves the highest score to EEPROM if the current score exceeds the previous high score.
playGivenMelody	- Plays a melody using the speaker, given an array of notes and durations.
startSequence	- Lights up LEDs sequentially at the start of the game for visual effect.
checkGameEnd	- Checks if the game should end based on the total number of missed moles.
twinkleLEDs	- Makes all LEDs twinkle for a specified duration, used for game-over effects.
resetGame	- Resets game variables, saves the highest score, displays reset messages, and restarts the game sequence.
updateSpeedLevel	- Adjusts the LED activation time based on the number of successful hits.
updateLCD	- Updates the LCD display with the current number of successful hits and remaining lives.
showHighestScore	- Displays the highest score on the LCD at the beginning of the game.

3.5 Results

Figure 4: game board

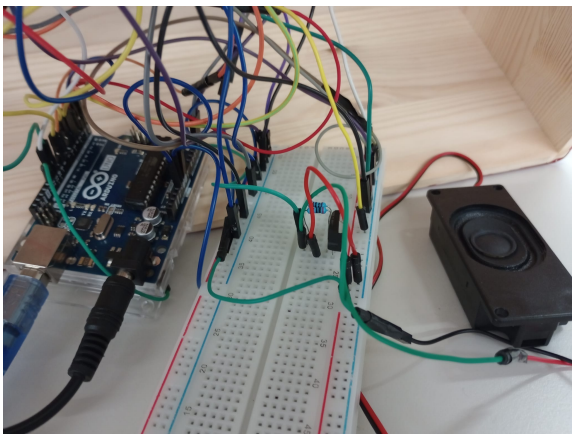

Figure 5: wiring

4 Conclusions

4.1 Possible Expansions

Audio improvement: To make the game more immersive, an audio module could be attached to play back music from mp3 files throughout the whole game rather than just sound effects.
Traditional physical movement: Although there are implementations of the whack-a-mole game just as ours, when one thinks of the game as a concept the physical movement of the moles going up when they are "whackable" comes to mind. This can be implemented by programming and implementing a physical system that would cause the up and down movement.
Game difficulty: As of now, one mole pops up after the other, though added difficulty could be implemented by having some appear very close together or even simultaneously.
Robustness: The TIP120 transistor attached to the speaker would ideally require a capacitor to be suitable for long-term use.

4.2 Lessons Learned

Diligence in testing every component in isolation as well as their interplay together (Testing every button, the speaker and LCD, testing the code in simulation vs. with the hardware, etc.) paid off for the project and was vital in bringing it to completion. Without taking the time initially to do this or implementing every element in order of their importance the course of the project would've run far more inefficiently. For this, making use of Arduino UNOs available to us as well as other materials to cross-check made things easier. Thus, utilizing all means possible to us became an useful strategy, widely applicable in future projects.

5 Appendix

Links to Amazon listings of hardware bought:
Buttons, Speaker, TIP120 Transistor
Link to source code on github: source code
ChatGPT: for partial text polishing and code debugging tips.