Culminating Activity Technical Report

TEJ4M1 06/16/2025

Arduino-Powered Game Console

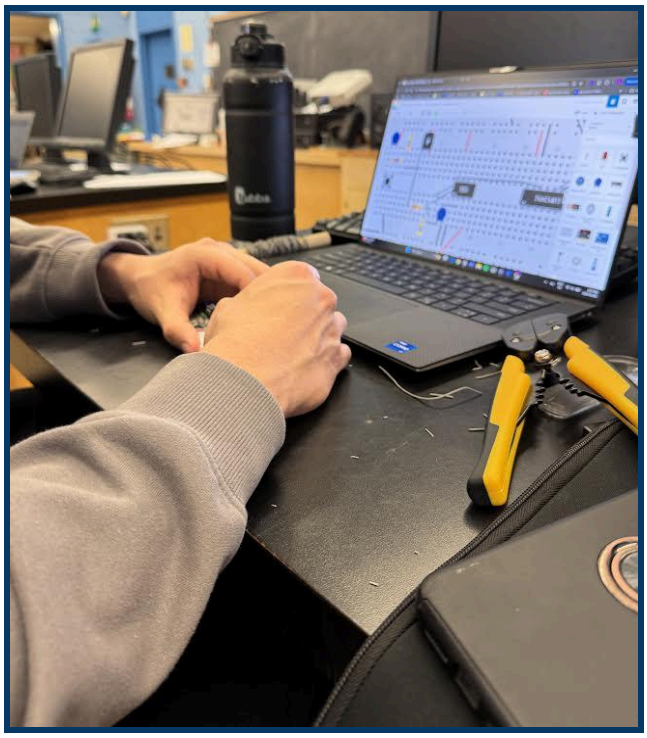
Jeevan Sanchez, Lakshya Sharma

Ms. Goczi

Table of Contents

1. Introduction

Memory games are a fun way to challenge the brain. To capture their essence, we built a game console featuring two memory games—Whack-a-Light and Simon Says—for our culminating project. The console is powered and programmed with an ELEGOO Arduino Uno microcontroller, utilizing LED pushbuttons and a buzzer to create the game effects.



2. Equipment List

The tools we used to fabricate our project.

Equipment Used	Function
Standard Breadboard	Board to house circuit.
Jumper Wire	Electrical connectivity.
Wire Stripper	Trim wires.
Multimeter	Test electrical connectivity.

3. Parts List

Quantity	Component	Function
1	Arduino Uno	Main controller and programmer.
6	LED Momentary Pushbuttons	Input buttons with LED feedback.
5	100-ohm Resistors (LED Buttons)	Limit current to LEDs.
1	OSEPP Push Button Module	Additional input button.
1	DC Buzzer	Sound output for feedback.
1	1000-ohm Resistor (Buzzer)	Limits current to buzzer.
1	Breadboard	Prototype circuit connections.

4. Circuit Operation

The circuit is powered and controlled by an ELEGOO Arduino Uno, which manages input from pushbuttons, output to LEDs and a buzzer, and handles game logic in code. The system includes five LED-button pairs, a toggle switch for game selection, and a passive DC-buzzer for audio feedback.

Power and Initialization

The Arduino is powered via USB. Upon startup, it configures all LED pins as outputs and button pins as inputs with internal pull-up resistors using `INPUT_PULLUP`. The buzzer pin is also set as an output to allow tone generation using PWM.

Game Selection and Control

A toggle switch is used to activate the console and select between game modes. When the system detects a press:

- Release of switch on blue LED starts the **Whack-a-Light** game.
- Release of switch on green LED starts the **Simon Says** game.

A brief delay is used to count how many times the switch was pressed before launching a mode.

Inputs and Outputs

Each button is paired with an LED of the same color. Players interact with the console by pressing these buttons in response to the lit LEDs. The Arduino reads button states in real time and compares them to expected inputs based on game logic.

- LEDs are controlled using `digitalWrite(ledPin, HIGH/LOW)`.
- Buttons are checked with `digitalRead(buttonPin)`.
- The buzzer provides feedback using `tone(pin, frequency, duration)`.

Wiring

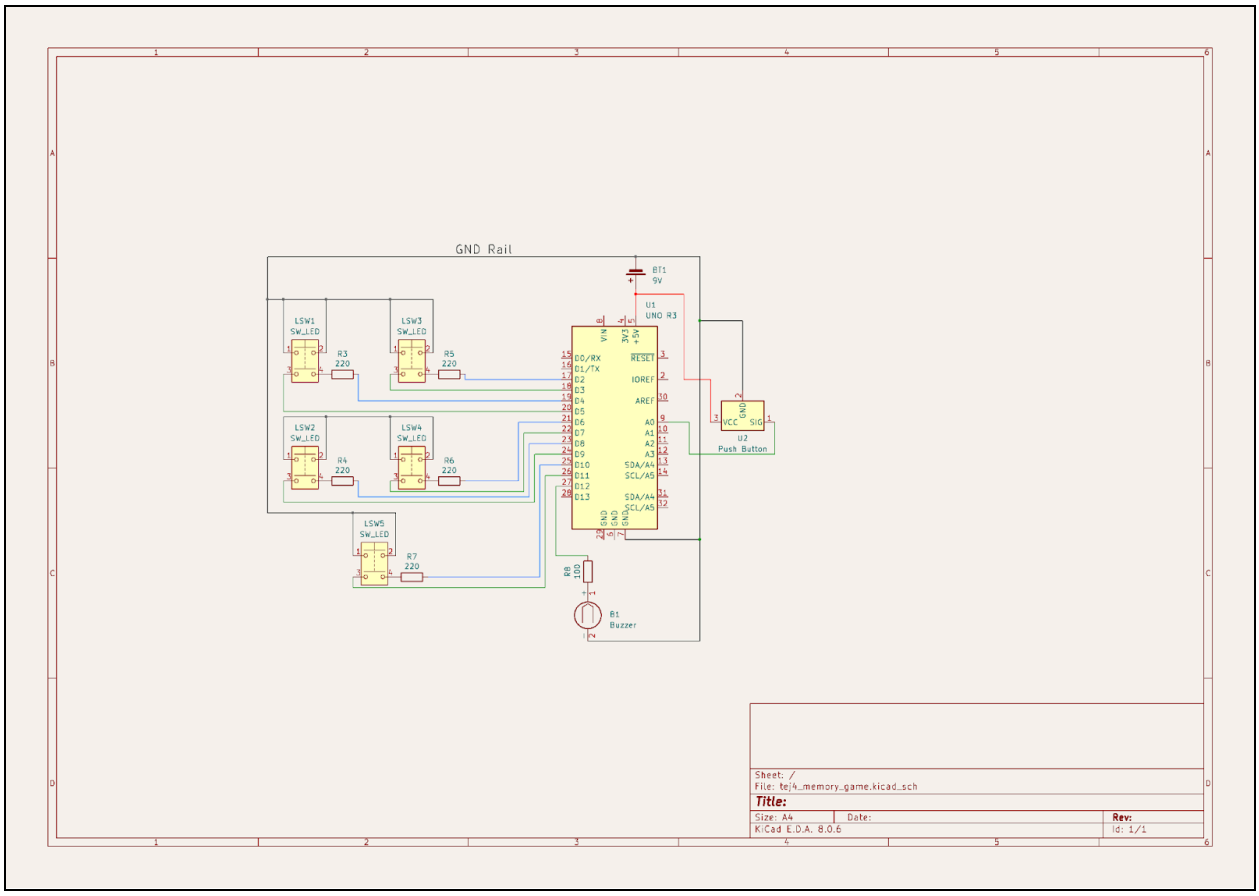
All components were connected on a solderless breadboard using jumper wires. Each LED is connected to a digital output pin through a 100-ohm resistor to limit current. The momentary pushbuttons are connected to digital input pins and grounded on the opposite side, using the Arduino's internal pull-up resistors to maintain a default HIGH state.

The toggle switch is wired similarly to the pushbuttons. The passive buzzer is connected to a digital output pin through a 1000-ohm resistor.

Power and ground are distributed using the breadboard rails. Special care was taken during wiring to avoid loose connections and signal overlap, though the high number of components made wire organization a challenge.

5. Schematic Diagram

Designed with KiCad.



7. Conclusion

Throughout this project, we built on what we learned in Grade 11 Computer Engineering, applying those concepts to more advanced circuit design and programming. The hardest part was soldering and wiring—ensuring proper electrical connectivity was both time-consuming and tricky. Moving forward, we plan to improve the user experience by building a more solid structure, adding better support, and upgrading the wiring for reliability, possibly by designing a custom PCB.