**Random Dice Roller with OLED**

**📘 Project Title:**

**OLED Dice Roller with Angular Animation, Screensaver, and Serial Monitor**

**🔧 Components Used**

**(**[**URL:-https://app.cirkitdesigner.com/project/4701fc8e-6df0-4dbc-aaeb-de212902bd25**](URL:-https://app.cirkitdesigner.com/project/4701fc8e-6df0-4dbc-aaeb-de212902bd25)**)**

| **Component Name** | **Pin No** | **Destination Component** | **Pin No** | **Special Remark** |
| --- | --- | --- | --- | --- |
| ESP32 Dev Module | GPIO14 | Push Button | 1 (One leg) | Button input trigger |
| ESP32 Dev Module | GPIO25 | Buzzer | +ve | Active-high buzzer |
| ESP32 Dev Module | GPIO26 | LED | +ve | Visual feedback |
| ESP32 Dev Module | 3.3V | OLED Display (VCC) | VCC | Power supply |
| ESP32 Dev Module | GND | OLED Display (GND) | GND | Ground connection |
| ESP32 Dev Module | GPIO22 | OLED Display (SCL) | SCL | I2C Clock line |
| ESP32 Dev Module | GPIO21 | OLED Display (SDA) | SDA | I2C Data line |

**📘 Code:-**

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 64

#define OLED\_RESET    -1

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET);

// Pins

#define BUTTON\_PIN 14

#define BUZZER\_PIN 25

#define LED\_PIN    26

bool buttonPressed = false;

unsigned long lastInputTime = 0;

bool inScreensaver = false;

const int idleTimeout = 15000;

// 8x8 star bitmap for screensaver

const unsigned char starBitmap[] PROGMEM = {

  0b00011000,

  0b00111100,

  0b01111110,

  0b11111111,

  0b01111110,

  0b00111100,

  0b00011000,

  0b00000000

};

// Dice dot layout (3x3) - true = dot present

bool diceDots[6][9] = {

  {0,0,0, 0,1,0, 0,0,0},  // 1

  {1,0,0, 0,0,0, 0,0,1},  // 2

  {1,0,0, 0,1,0, 0,0,1},  // 3

  {1,0,1, 0,0,0, 1,0,1},  // 4

  {1,0,1, 0,1,0, 1,0,1},  // 5

  {1,0,1, 1,0,1, 1,0,1}   // 6

};

// Draw rotated square (approximated angular effect)

void drawRotatedSquare(int cx, int cy, int size, bool rotate) {

  int half = size / 2;

  if (!rotate) {

    display.drawRect(cx - half, cy - half, size, size, WHITE);

  } else {

    // Rotate corners by 45° approx (square to diamond)

    int x0 = cx;

    int y0 = cy - half;

    int x1 = cx + half;

    int y1 = cy;

    int x2 = cx;

    int y2 = cy + half;

    int x3 = cx - half;

    int y3 = cy;

    display.drawLine(x0, y0, x1, y1, WHITE);

    display.drawLine(x1, y1, x2, y2, WHITE);

    display.drawLine(x2, y2, x3, y3, WHITE);

    display.drawLine(x3, y3, x0, y0, WHITE);

  }

}

// Draw single dice face with optional rotation

void drawDiceFace(int number, bool rotate = false) {

  display.clearDisplay();

  int diceSize = 40;

  int cx = SCREEN\_WIDTH / 2;

  int cy = SCREEN\_HEIGHT / 2;

  // Draw square or rotated border

  drawRotatedSquare(cx, cy, diceSize, rotate);

  // Dot spacing and radius

  int spacing = 12;

  int dotRadius = 3;

  int dx[] = {-spacing, 0, spacing};

  int dy[] = {-spacing, 0, spacing};

  for (int i = 0; i < 9; i++) {

    if (diceDots[number - 1][i]) {

      int row = i / 3;

      int col = i % 3;

      int x = cx + dx[col];

      int y = cy + dy[row];

      // Apply 45° rotation (approximation)

      if (rotate) {

        int temp = x;

        x = cx + (y - cy);

        y = cy + (temp - cx);

      }

      display.fillCircle(x, y, dotRadius, WHITE);

    }

  }

  display.display();

}

// Screensaver

void showScreensaver() {

  static int yOffset = -8;

  display.clearDisplay();

  for (int i = 0; i < 6; i++) {

    int x = 10 + i \* 20;

    int y = (yOffset + i \* 7) % SCREEN\_HEIGHT;

    display.drawBitmap(x, y, starBitmap, 8, 8, WHITE);

  }

  display.display();

  yOffset += 2;

  if (yOffset > SCREEN\_HEIGHT) yOffset = -8;

  delay(50);

}

// Rolling animation with border rotation

void animateDiceRoll() {

  Serial.println("Rolling dice...");

  digitalWrite(LED\_PIN, HIGH);

  digitalWrite(BUZZER\_PIN, HIGH);

  for (int i = 0; i < 10; i++) {

    int temp = random(1, 7);

    bool rotate = (i % 2 == 1);  // Alternate rotation

    drawDiceFace(temp, rotate);

    delay(100);

  }

  digitalWrite(LED\_PIN, LOW);

  digitalWrite(BUZZER\_PIN, LOW);

}

// Welcome screen

void showWelcomeScreen() {

  display.clearDisplay();

  display.setTextSize(1);

  display.setTextColor(WHITE);

  for (int y = -8; y < SCREEN\_HEIGHT + 8; y += 2) {

    display.clearDisplay();

    for (int i = 0; i < 6; i++) {

      int x = 10 + i \* 20;

      int yOffset = (y + i \* 7) % SCREEN\_HEIGHT;

      display.drawBitmap(x, yOffset, starBitmap, 8, 8, WHITE);

    }

    display.display();

    delay(50);

  }

  display.clearDisplay();

  display.setCursor(20, 25);

  display.println("  Welcome to");

  display.setCursor(20, 40);

  display.println("  Dice Roller");

  display.display();

  delay(3000);

  display.clearDisplay();

  display.display();

}

void setup() {

  pinMode(BUTTON\_PIN, INPUT);

  pinMode(BUZZER\_PIN, OUTPUT);

  pinMode(LED\_PIN, OUTPUT);

  Serial.begin(115200);

  Serial.println("Dice Roller starting...");

  if (!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) {

    Serial.println("OLED not found");

    while (1);

  }

  randomSeed(analogRead(0));

  showWelcomeScreen();

  drawDiceFace(1);

  lastInputTime = millis();

}

void loop() {

  bool state = digitalRead(BUTTON\_PIN);

  if (inScreensaver && state == HIGH) {

    Serial.println("Exiting screensaver...");

    inScreensaver = false;

    drawDiceFace(1);

    delay(300);

  }

  if (inScreensaver) {

    showScreensaver();

    return;

  }

  if (state == HIGH && !buttonPressed) {

    buttonPressed = true;

    lastInputTime = millis();

    animateDiceRoll();

    int result = random(1, 7);

    drawDiceFace(result);

    Serial.print("Final Dice Face: ");

    Serial.println(result);

  }

  if (state == LOW && buttonPressed) {

    buttonPressed = false;

  }

  if (!buttonPressed && millis() - lastInputTime > idleTimeout) {

    Serial.println("Entering screensaver...");

    inScreensaver = true;

  }

}

**📄 Functional Overview**

This project is a **digital dice roller** using an **OLED display (128x64)** to simulate dice faces (1–6) in **3x3 binary dot format**, inside a **rotating square** representing the dice border. It includes:

* **Button-triggered roll** with angular animation.
* **Random face output** displayed.
* **LED and buzzer feedback** during roll.
* **Screensaver mode** after inactivity.
* **Serial Monitor** logs actions and results.

**🎯 Features**

* ✅ Dice face displayed in classic dot pattern.
* 🔁 Angular rotation effect during roll (diamond tilt simulation).
* 🎲 Random dice number (1–6) on button press.
* 🎇 Screensaver activates after 15 seconds idle.
* 🔊 Buzzer & 💡 LED during rolling.
* 🖥 Serial Monitor logs:
  + Dice start
  + Final face
  + Screensaver entry/exit

**🧠 How It Works**

**🕹 Button Interaction**

* Pressing the button initiates a **rolling animation** (10 quick dice faces).
* The final result is a **randomly selected dice face (1–6)**.

**🔄 Animation Effect**

* During roll: dice face alternates between **square and rotated diamond**.

**⏳ Idle Screensaver**

* After **15 seconds** of no interaction, screen shows **moving stars**.
* Exits screensaver immediately on button press.

**💬 Serial Monitor Output**

* Helps **debug** and **track activity** (roll start, final result, idle timeout).

**🧪 Example Serial Output**

Dice Roller starting...

Rolling dice...

Final Dice Face: 4

Entering screensaver...

Exiting screensaver...

Rolling dice...

Final Dice Face: 6

**📦 Required Libraries**

Ensure you have the following libraries installed via Library Manager:

* Adafruit\_GFX
* Adafruit\_SSD1306
* Wire (default)

**🖥 Arduino IDE Settings**

| **Setting** | **Value** |
| --- | --- |
| Board | ESP32 Dev Module |
| Flash Frequency | 80 MHz (default) |
| Upload Speed | 115200 (recommended) |
| Partition Scheme | Default (4MB w/ SPIFSS) |
| Port | COMx (your ESP32 port) |

**🧾 Code Summary**

void setup() {

// Pin initialization

// OLED start

// Serial monitor setup

// Show welcome screen

}

void loop() {

// Button press logic

// Dice rolling animation

// Final face display

// Screensaver on idle

}

**📷 Display Behavior**

* **Dice face layout**: 3x3 dot pattern.
* **Dice face box**: switches between **square** and **rotated square (diamond)**.
* **Screensaver**: Animated stars moving downward diagonally.