**ESP32-Based Calculator with LCD Display**

**📄 Project Title:**

**ESP32-Based Calculator with LCD Display, Keypad, Buzzer & LED Feedback**

**🧾 Objective:**

To build a functional calculator using an ESP32 microcontroller with real-time input from a 4x4 keypad and output on a 20x4 I2C LCD. The system provides tactile and visual feedback using a buzzer and LED for every keypress and performs basic arithmetic operations: **addition, subtraction, multiplication, and division.**

**🧰 Components Used:**

**(URL:-** [**https://app.cirkitdesigner.com/project/36ecf600-58d1-42e4-895c-f56082117aad**](https://app.cirkitdesigner.com/project/36ecf600-58d1-42e4-895c-f56082117aad)**)**

| **Component Name** | **Pin No (ESP32)** | **Destination Component** | **Pin No** | **Special Remark** |
| --- | --- | --- | --- | --- |
| ESP32 Dev Module | - | - | - | Core microcontroller |
| I2C 20x4 LCD Display | GPIO 21 (SDA) | ESP32 | SDA | I2C interface |
|  | GPIO 22 (SCL) | ESP32 | SCL | I2C interface |
| 4x4 Keypad | GPIO 13 | Row 1 | R1 | Connect using male-female wires |
|  | GPIO 12 | Row 2 | R2 |  |
|  | GPIO 14 | Row 3 | R3 |  |
|  | GPIO 25 | Row 4 | R4 |  |
|  | GPIO 18 | Col 1 | C1 |  |
|  | GPIO 5 | Col 2 | C2 |  |
|  | GPIO 4 | Col 3 | C3 |  |
|  | GPIO 15 | Col 4 | C4 |  |
| Buzzer | GPIO 27 | GND | - | Active buzzer (optional resistor) |
| LED | GPIO 26 | GND | - | Any color |

**🔧 Circuit Diagram Overview:**

* **I2C LCD:** Connect SDA to GPIO21, SCL to GPIO22.
* **Keypad Rows & Columns:** Connected as per table.
* **Buzzer & LED:** Direct GPIO output pins, controlled in code.

**#include <Wire.h>**

**#include <LiquidCrystal\_I2C.h>**

**#include <Keypad.h>**

**// LCD Setup: 20x4 I2C (Try address 0x27 or 0x3F)**

**LiquidCrystal\_I2C lcd(0x27, 20, 4);  // Change to 0x3F if nothing appears**

**// Keypad Layout**

**char keys[4][4] = {**

**{'1', '2', '3', '+'},**

**{'4', '5', '6', '-'},**

**{'7', '8', '9', '\*'},**

**{'C', '0', '=', '/'}**

**};**

**// Safe GPIO Pins for ESP32**

**byte rowPins[4] = {13, 12, 14, 25};  // R1-R4**

**byte colPins[4] = {18, 5, 4, 15};    // C1-C4**

**Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, 4, 4);**

**#define BUZZER\_PIN 27**

**#define LED\_PIN 26**

**String input = "";**

**char operatorChar = '\0';**

**void setup() {**

**Serial.begin(115200);**

**pinMode(BUZZER\_PIN, OUTPUT);**

**pinMode(LED\_PIN, OUTPUT);**

**lcd.init();**

**lcd.backlight();**

**lcd.clear();**

**lcd.setCursor(0, 0);**

**lcd.print("Calculator Ready");**

**Serial.println("🔢 Calculator Ready. Use keypad.");**

**}**

**void loop() {**

**char key = keypad.getKey();**

**if (key) {**

**// Feedback**

**digitalWrite(LED\_PIN, HIGH);**

**digitalWrite(BUZZER\_PIN, HIGH);**

**delay(100);**

**digitalWrite(LED\_PIN, LOW);**

**digitalWrite(BUZZER\_PIN, LOW);**

**// Clear**

**if (key == 'C') {**

**input = "";**

**operatorChar = '\0';**

**lcd.clear();**

**lcd.setCursor(0, 0);**

**lcd.print("Cleared");**

**lcd.setCursor(0, 1);**

**lcd.print("Input: ");**

**Serial.println("🧹 Cleared");**

**return;**

**}**

**// Evaluate**

**else if (key == '=') {**

**lcd.setCursor(0, 1);**

**lcd.print("Input: " + input + "    ");**

**Serial.println("Input: " + input);**

**if (operatorChar != '\0') {**

**int opIndex = input.indexOf(operatorChar);**

**if (opIndex > 0 && opIndex < input.length() - 1) {**

**float num1 = input.substring(0, opIndex).toFloat();**

**float num2 = input.substring(opIndex + 1).toFloat();**

**float result = 0;**

**bool valid = true;**

**switch (operatorChar) {**

**case '+': result = num1 + num2; break;**

**case '-': result = num1 - num2; break;**

**case '\*': result = num1 \* num2; break;**

**case '/':**

**if (num2 != 0) result = num1 / num2;**

**else {**

**lcd.setCursor(0, 2);**

**lcd.print("  Divide by 0     ");**

**Serial.println("❌ Error: Divide by 0");**

**valid = false;**

**}**

**break;**

**}**

**if (valid) {**

**lcd.setCursor(0, 2);**

**lcd.print("Result: ");**

**lcd.print(result);**

**Serial.print("✅ Result: ");**

**Serial.println(result);**

**}**

**} else {**

**lcd.setCursor(0, 2);**

**lcd.print("⚠ Format error     ");**

**Serial.println("⚠️ Format error. Try 7\*8=");**

**}**

**} else {**

**lcd.setCursor(0, 2);**

**lcd.print("⚠ No operator!     ");**

**Serial.println("⚠️ No operator found!");**

**}**

**input = "";**

**operatorChar = '\0';**

**// ❌ Removed delay + clear, keep result until next keypress**

**}**

**// Input keys**

**else {**

**if (key == '+' || key == '-' || key == '\*' || key == '/') {**

**if (operatorChar == '\0') {**

**operatorChar = key;**

**input += key;**

**} else {**

**lcd.setCursor(0, 3);**

**lcd.print("⚠ 1 operator only ");**

**Serial.println("⚠️ Operator already used.");**

**delay(1500);**

**lcd.setCursor(0, 3);**

**lcd.print("                    ");**

**}**

**} else if (key >= '0' && key <= '9') {**

**input += key;**

**}**

**// Display current input**

**lcd.setCursor(0, 0);**

**lcd.print("You pressed: ");**

**lcd.print(key);**

**lcd.print("     "); // padding**

**lcd.setCursor(0, 1);**

**lcd.print("Input: " + input + "     ");**

**Serial.print("You pressed key: ");**

**Serial.println(key);**

**Serial.println("Input: " + input);**

**}**

**}**

**}**

**💻 Functionality Overview:**

* **Startup:** Displays "Calculator Ready" on LCD.
* **Key Input:** Takes input via the keypad. Digits (0-9) and operations (+, −, ×, ÷) are supported.
* **Feedback:** LED and buzzer blink once on every keypress.
* **Display:** Shows the input string and result on a 20x4 LCD.
* **Clear:** Press 'C' to reset input.
* **Evaluate:** Press '=' to compute result.
* **Errors Handled:**
  + Multiple operators are prevented.
  + Divide-by-zero warning.
  + Format errors displayed (e.g., missing operand).
* **Persistence:** The result remains on the screen until a new key is pressed (no auto-clear).

**🔢 Keypad Mapping (4x4):**

| 1 | 2 | 3 | + |

| 4 | 5 | 6 | - |

| 7 | 8 | 9 | \* |

| C | 0 | = | / |

**📟 LCD Display Format:**

| **Line** | **Content** |
| --- | --- |
| 1 | You pressed: X |
| 2 | Input: 8\*9 |
| 3 | Result: 72 (after '=') |
| 4 | Warning/Status messages if needed |

**🔉 User Feedback:**

* **Buzzer**: Beeps on keypress.
* **LED**: Flashes momentarily with each key.
* **LCD**: Shows current key, input expression, and result or error.

**🧠 How It Works (Logic Flow):**

1. **Initialize LCD, keypad, buzzer, LED**
2. Wait for keypress:
   * If digit/operator → Add to input string
   * If '=' → Parse and evaluate
   * If 'C' → Clear input and screen
3. **Evaluate on '='**:
   * Identify the operator
   * Split the input into operands
   * Perform arithmetic
   * Handle divide-by-zero or invalid format
4. Show result and wait for the next input.

**🧪 Testing Suggestions:**

Try the following inputs:

* 5+3=, 7\*8=, 9-2=, 8/2=
* Press = without any input → Format error
* Try 9/0= → Divide by zero error
* Use C to clear

**📦 Libraries Required:**

Install these from the Arduino Library Manager:

* LiquidCrystal\_I2C by Frank de Brabander or Marco Schwartz
* Keypad by Mark Stanley & Alexander Brevig

**🧾 Serial Monitor Output (Optional Debug):**

🔢 Calculator Ready. Use keypad.

You pressed key: 7

Input: 7

You pressed key: \*

Input: 7\*

You pressed key: 8

Input: 7\*8

✅ Result: 56