Hardware Experiment of Scientific Calculator

Murra Rajesh Kumar Reddy - EE24BTECH11043

1. Introduction

This project involves designing a calculator using an Arduino Uno, a 16x2 LCD display, a 4x4 keypad, and a shift button to toggle between basic arithmetic and scientific operations. The objective is to create a user-friendly calculator with an interactive interface.

2. Components Used

The hardware components used in this project include:

1) **Arduino Uno** - Microcontroller for processing inputs and performing calculations.
2) **16x2 LCD Display** - Displays user inputs and computed results.
3) **4x4 Keypad** - Allows user input for numbers and mathematical operations.
4) **Push Button** - Used to shift between basic and scientific functions.
5) **Resistors and Wires** - Supporting electronic components.

3. Working Principle

3-A. User Input

The user enters numbers and operations via the 4x4 keypad.

3-B. Processing

The Arduino reads input values and processes calculations accordingly.

3-C. Shift Functionality

A button is used to toggle between basic arithmetic functions (+, -, ×, /) and scientific functions (sin, cos, tan, log, etc.).

3-D. Output Display

The computed result is displayed on the LCD screen.

4. Features

1) Performs basic arithmetic calculations: addition, subtraction, multiplication, and division.
2) Supports scientific functions such as trigonometric and logarithmic calculations.
3) User-friendly interface with a clear LCD display.
4) The shift button enables additional functionalities without extra hardware.

5. Circuit Pin Connections

LCD Pin	Function	Arduino Pin	Notes
RS	Register Select	D8 (PB0)	LOW: Command; HIGH: Data
E	Enable	D9 (PB1)	Latches data
D4	Data Bit 4	D10 (PB2)	4-bit interface
D5	Data Bit 5	D11 (PB3)	4-bit interface
D6	Data Bit 6	D12 (PB4)	4-bit interface
D7	Data Bit 7	D13 (PB5)	4-bit interface

Additional connections: VSS (GND), VDD (+5V), VO (potentiometer for contrast), RW (GND), A (+5V backlight), K (GND).

1) Push Button Connections

Button	Function	Arduino Pin	Notes
Digits 0–5	Enter digits 0–5	D2, D3, D4, D5, D6, D7	Active-low; internal pull-ups
Digits 6–9	Enter digits 6–9	A0, A1, A2, A3	Digital inputs; pull-ups enabled
Clear (C)	Clear input	A4	Resets expression
Enter (=)	Evaluate expression	A5	Triggers computation
Extra Button 1	Additional function	(Assign as needed)	
Extra Button 2	Additional function	(Assign as needed)	

Note: The extra two push buttons can be used for further functions or for shift modes if needed.

6. Code Structure

The code follows a modular approach, with key sections handling different functionalities:

6-A. Library Inclusions

The necessary libraries are included at the beginning to interface with the LCD and perform mathematical operations.

Listing 1. Library Inclusion

```
#include <LiquidCrystal.h>   // LCD display control
#include <math.h>            // Mathematical functions (sin, cos, etc.)
```

6-B. Hardware Setup

The LCD display is connected to the Arduino through digital pins, and the push buttons are assigned to specific pins for input handling. A potentiometer is used for selecting different functions.

Listing 2. Hardware Setup

```
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);  // LCD pins
const int potPin = A0;  // Potentiometer for function selection
const int buttonPins[] = {2, 3, 4, 5, 6}; // Buttons for input
```

6-C. Input Handling

The push buttons allow users to enter numbers and select operations. The potentiometer maps to different scientific functions.

Listing 3. Reading Button Input

```
void readButtons() {
    for (int i = 0; i < 5; i++) {
        if (digitalRead(buttonPins[i]) == LOW) {
            // Capture input value
        }
    }
}
```

Listing 4. Mapping Potentiometer Input

```
int getFunction() {
    int value = analogRead(potPin);
    return map(value, 0, 1023, 0, 5);  // Example: 0 = sin, 1 = cos, etc.
}
```

6-D. Performing Calculations

A function processes user input and performs the required mathematical operation.

Listing 5. Calculation Function

```
float calculate(float num1, float num2, char op) {
    switch (op) {
        case '+': return num1 + num2;
        case '-': return num1 - num2;
        case '*': return num1 * num2;
        case '/': return (num2 != 0) ? num1 / num2 : 0;
        case 's': return sin(num1);  // Example: Scientific function
        default: return 0;
    }
}
```

6-E. Displaying Output

Results are displayed on the **16x2 LCD screen** after processing.

Listing 6. LCD Output Display

```
void displayResult(float result) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Result:");
    lcd.setCursor(0, 1);
    lcd.print(result);
}
```

6-F. Main Loop Execution

The loop continuously checks for user input, processes calculations, and updates the LCD display.

Listing 7. Main Loop Execution

```
void loop() {
    int selectedFunction = getFunction();
    readButtons();
    // Perform calculation and display result
}
```

7. Features of the Code

1) Supports **basic arithmetic** operations (+, -, ×, /)
2) Includes **scientific functions** (sin, cos, log, etc.)
3) Uses a **potentiometer** for function selection
4) Displays results on an **LCD screen**
5) Handles **multiple inputs** via push buttons

8. Challenges and Solutions

1) **Keypad Debouncing:** Implemented software techniques to handle multiple keypresses correctly.
2) **Limited LCD Display Space:** Optimized data display by clearing and refreshing necessary portions.
3) **Memory Constraints:** Efficiently managed code structure to ensure smooth operation.

9. Conclusion

This project successfully demonstrates the implementation of a calculator using an Arduino. It serves as an educational tool for understanding microcontroller-based projects and user interface design.