Building an Arduino Calculator with Breadboard and Push Buttons

Niketh Achanta - EE24BTECH11047

1 Introduction

This report details the process of building a functional calculator using an Arduino microcontroller, a breadboard, push buttons, and an LCD display. The calculator can perform basic arithmetic operations (addition, subtraction, multiplication, and division) as well as trigonometric functions (sine, cosine, tangent, and their inverse functions). This project demonstrates fundamental concepts in embedded systems programming, user interface design, and mathematical computation.

2 Materials Required

- Arduino board (UNO)
- Breadboard
- 16x2 LCD display
- Push buttons (at least 14)
- Jumper wires
- Potentiometer (For LCD brightness contrast)

3 Circuit Design

The calculator circuit consists of three main components:

3.1 LCD Display

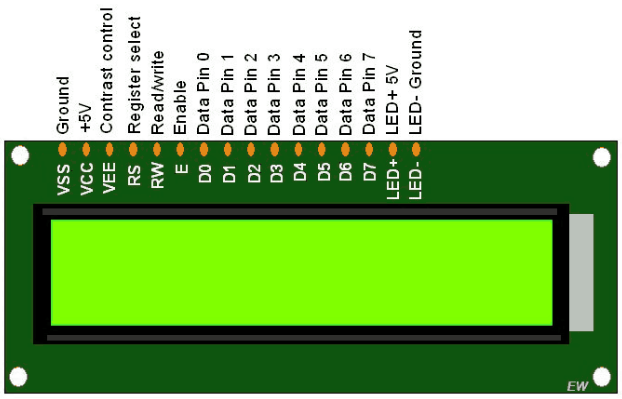

Figure 1: LCD Display Pinout

The 16x2 LCD display is connected to the Arduino in 4-bit mode to save pins. The connections are as follows:

- RS (Register Select) pin connected to Arduino pin 1 (PD1)
- E (Enable) pin connected to Arduino pin 0 (PD0)
- D4 connected to Arduino pin 5 (PD5)
- D5 connected to Arduino pin 4 (PD4)
- D6 connected to Arduino pin 3 (PD3)
- D7 connected to Arduino pin 2 (PD2)

3.2 Push Buttons

The calculator uses multiple push buttons for input:

- Numeric buttons (0-9) connected to Arduino pins 13 (PB5) through 8 (PB0) and pins 7 (PD7), 6 (PD6), A0 (PC0), and A1 (PC1)
- Operator button (cycles through +, -, *, /) connected to Arduino pin A2 (PC2)
- Trigonometric function button (cycles through sin, cos, tan, asin, acos, atan) connected to Arduino pin A3 (PC3)
- Decimal point button connected to Arduino pin A4 (PC4)
- Enter/equals button connected to Arduino pin A5 (PC5)

4 Wiring

The wiring should be done as follows:

1. Connect the LCD pins as specified in the LCD Display section
2. For each button:
    - Connect one terminal to the corresponding Arduino pin
    - Connect the other terminal to ground
3. Connect the Arduino's 5V and GND pins to the breadboard's power rails

5 Code Explanation

This section provides a detailed explanation of the scientific calculator implementation for an AVR microcontroller.

5.1 Overview

The code implements a scientific calculator on an AVR microcontroller with an LCD display and a button interface. It supports basic arithmetic operations and trigonometric functions. The calculator allows users to input numeric values, select operations, and compute results.

5.2 Hardware Configuration

- **LCD Interface:** Connected to PORTB with RS pin on PB0 and EN pin on PB1
- **Button Interface:**
  - Numeric buttons (0-7): Connected to PORTD pins PD0-PD7
  - Numeric buttons (8-9): Connected to PORTC pins PC0-PC1
  - Decimal point button: Connected to PC2
  - Operation selection button: Connected to PC3
  - Trigonometric function button: Connected to PC4
  - Enter button: Connected to PC5

5.3 Key Components

5.3.1 LCD Control Functions

- `lcd_init()`: Initializes the LCD in 4-bit mode
- `lcd_cmd()`: Sends command instructions to the LCD
- `lcd_data()`: Sends character data to display on the LCD
- `lcd_print()`: Outputs a string to the LCD

5.3.2 Button Handling

- `init_buttons()`: Configures button pins as inputs with pull-up resistors
- `check_button()`: Detects button presses with debouncing logic

5.3.3 Expression Handling

- `append_char()`: Adds a character to the expression buffer
- `update_lcd()`: Refreshes the LCD with the current expression
- `append_operator()`: Handles addition of arithmetic operators (+, -, *, /)
- `append_trig_function()`: Handles addition of trigonometric functions
- `process_enter()`: Placeholder for expression evaluation

5.4 Program Flow

The main program follows this execution flow:

1. Initialize the expression buffer, LCD, and button interface
2. Display a welcome message prompting the user to enter an expression
3. Enter the main loop that continuously monitors button presses:
   - Decimal point button adds a decimal point to the current number
   - Numeric buttons (0-9) append digits to the expression
   - Operation button cycles through arithmetic operators
   - Trigonometric button cycles through trig functions
   - Enter button processes the expression (evaluation logic is a placeholder)

5.5 Features and Behavior

- Automatically inserts a leading zero when adding a decimal point after an operator
- Prevents multiple decimal points within a single number
- Supports nesting of functions (e.g., sin(cos(x)))
- Clears the display when entering a new expression after a result
- Cycles through operators (+, -, *, /) and trigonometric functions (sin, cos, tan, asin, acos, atan)

5.6 Implementation Considerations

- The code uses a fixed-size buffer (`MAX_EXPR_LENGTH = 32`) to store the expression
- Debouncing is implemented with delay functions
- The actual expression evaluation logic is not implemented in the current version
- Button state tracking prevents duplicate entries and ensures proper syntax

6 Mathematical Implementation

The calculator implements mathematical operations using a recursive descent parser in the `evaluate_expression()` function. This approach allows it to handle complex expressions with nested parentheses and functions.

For trigonometric functions, the calculator converts between degrees and radians:

- For direct trig functions (sin, cos, tan), it converts the input from degrees to radians using the formula
$$\text{radians} = \text{degrees} \times \pi/180$$
- For inverse trig functions (asin, acos, atan), it converts the output from radians to degrees using
$$\text{degrees} = \text{radians} \times 180/\pi$$

The calculator also handles special cases like division by zero, which returns NaN (Not a Number).

7 Conclusion

This Arduino calculator project demonstrates how to build a functional calculator using basic electronic components and programming techniques. The implementation includes:

- Hardware interfacing with an LCD display and push buttons
- User input processing and display
- Mathematical expression parsing and evaluation
- Error handling for cases like division by zero

The project can be extended in several ways, such as adding more mathematical functions, improving the user interface, or adding memory functions.

8 References

1. Code By MBS Aravind
2. AI suggestions for connections and other hardware analysis
3. Stock pictures for pinout diagrams