Scientific Calculator

EE24BTECH11065 - Spoorthi yellamanchali

1 Aim

To build a Scientific calculator.

2 Hardware Components

The main components used in this project are listed in Table ??.

Component	Quantity
Arduino Microcontroller	1
Non-I2C 16x2 LCD	1
Push Buttons (0-9 digits)	10
Push Buttons (+, -, *, /)	4
Push Button (Scroll for sin, cos, tan)	1
15kΩResistors	10
1kΩResistors	1
Breadboard	1
Jumper Wires	As required

TABLE 0: List of Components

3 Circuit Connections

The LCD is operated in 4-bit mode to reduce the number of control pins. The key connections are given in Table ??.

LCD Pin	Connection
Vss	GND
Pin 2	5V
Pin 3	1kΩresistor to GND
Pin 4	Arduino Pin 2
Pin 5	Connected to all push buttons
Pin 6	Arduino Pin 3
Pin 15	1kΩresistor to 5V
Pin 16	GND

TABLE 0: LCD Pin Connections

4 Push buttons

Button Function	**Arduino Connection**
Digit 0-9 Selection	A0 (Analog Read)
Addition (+)	Digital Pin 13
Subtraction (-)	Digital Pin 12
Multiplication (*)	Digital Pin 11
Division (/)	Digital Pin 10
Open Bracket (()	Digital Pin 9
Close Bracket ())	A3 (Analog Read)
Logarithm (ln/log)	A4 (Analog Read)
Evaluate (Calculate/Enter)	Digital Pin 8
Decimal Point (.)	Digital Pin 1
Delete (Backspace)	Digital Pin 0

TABLE 0: Button Connections for Scientific Calculator

5 Analog-to-Digital Conversion (ADC) Values

Analog-to-Digital Conversion (ADC) is a technique used to convert an analog signal into a digital value that can be processed by a microcontroller. In this project, ADC values are used to read multiple button inputs using a single analog pin. This is achieved by connecting different resistors to each button, creating a unique voltage divider network. When a button is pressed, a specific voltage is applied to the analog pin, which the microcontroller reads and maps to a corresponding digital value.

The main advantages of using ADC values for button inputs include:

- Reduction in the number of required input pins, optimizing microcontroller resources.
- Simplified wiring, reducing circuit complexity and improving reliability.
- Efficient and accurate button detection using predefined voltage thresholds.

The ADC technique allows multiple buttons to be read through a single analog pin, saving digital input pins. Arithmetic operations are handled through separate digital pins, and trigonometric functions are accessed using a single button with a scrolling method

6 Software Implementation

The firmware is developed in Embedded C and compiled using AVR-GCC. The main functions include:

- Reading ADC values for digit buttons
- Detecting arithmetic operations using digital pins
- Implementing a scroll method for trigonometric functions
- Displaying input and results on the LCD
- Performing calculations and updating the display dynamically

7 Advantages of AVR-GCC

AVR-GCC (GNU Compiler Collection for AVR) offers several benefits:

- Open-source and free, making it accessible for all developers.
- Efficient optimization techniques that reduce program size and execution time.
- Portability across different AVR microcontrollers.
- Extensive library support for handling hardware interfaces such as LCD and ADC.
- Compatibility with various development environments, including Atmel Studio and Arduino IDE.

8 Conclusion

This project successfully demonstrates the implementation of a scientific calculator using an Arduino microcontroller. By efficiently utilizing ADC values, digital pins, and a non-I2C LCD, the design minimizes hardware complexity while maintaining functionality. The use of AVR-GCC enhances performance and flexibility, making the project a practical and cost-effective solution for embedded applications.