

Project Documentation:

PC Console Application for Arduino

Based Serial Calculator

Table of Contents

Introduction	1
Objectives.....	1
System Architecture	1
Components:	1
Workflow:	1
UML Diagram	1
Arduino Software Structure.....	2
Features.....	2
Technologies Used	2
How It Works.....	2
Example Usage	8
Error Handling.....	9
Testing the Setup.....	10
Note	11
Bibliography.....	11

List of Figures

Figure 1 UML Diagram	1
Figure 2 Changing directory	3
Figure 3 Create an executable	4
Figure 4 Running the executable.....	4
Figure 5 Alt. running the executable	5
Figure 6 Importing SerialPort.cpp	5
Figure 7 Importing SerialPort.h	5
Figure 8 Build on Visual Studio	6
Figure 9 Running executable from Visual Studio	6
Figure 10 Selecting the board and port.....	7
Figure 11 Selecting the board and port extended	7
Figure 12 Upload the code to the microcontroller	8
Figure 13 Usage example 1	8
Figure 14 Usage example 2	9
Figure 15 When Arduino is not connected	9
Figure 16 Invalid input format.....	10
Figure 17 Division by zero error.....	10
Figure 18 Failed to connect error	11

Introduction

This is a documentation describing the console application. The console application serves as a user interface between the user and the microcontroller, and the microcontroller performs basic arithmetic operations. The microcontroller used in this project is an Arduino Uno, however, this is compatible with Arduino in as much as the code is uploaded to the choice of microcontroller.

The Arduino program acts as the processing unit for a serial-based calculator system.

The user enters the expression via the console; this is then sent over to the Arduino over a serial connection. The Arduino performs the calculation and returns the result, which is then displayed in the console.

GitHub Link to Project: <https://github.com/Mexperse/Serial-Calculator.git>

Objectives

- Accept basic arithmetic expressions: +, -, *, / from the user (e.g., 2.5 * 5).
- Establish and manage serial communication with an Arduino board.
- Send expressions to Arduino and receive computed results.
- Display results or errors.

System Architecture

Components:

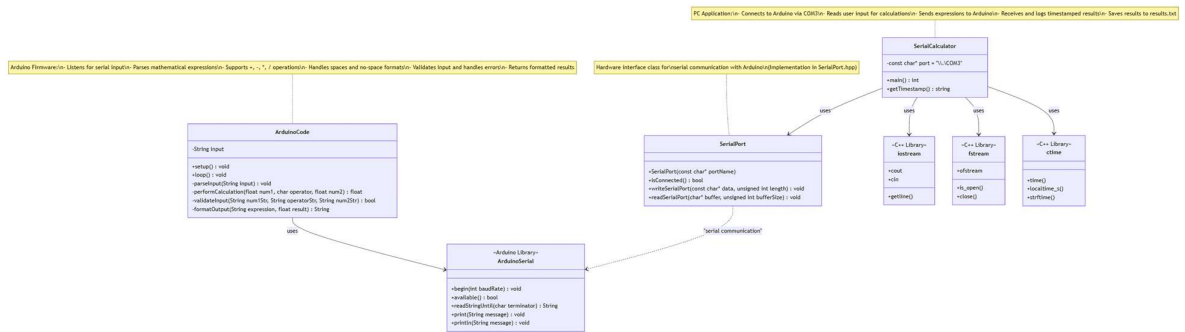
- PC console application (written in C++)
- Arduino microcontroller (Arduino Uno in this case!)
- Serial communication (USB via virtual COM port)

Workflow:

User ➡ Console App ➡ Serial Port ➡ Arduino ➡ Serial Port ➡ Console App ➡ User

UML Diagram

Figure 1 UML Diagram



Source: Generated using the prompt “Can you generate a UML diagram based on this code”

Arduino Software Structure

Function	Description
setup()	Initializes the serial communication
loop()	Continuously checks for incoming expressions, processes them, and returns results
switch(arithOperator)	Checks the operator and performs a set of operations based on that

Features

- Cross-platform CLI
- Serial port
- Input validation (format: `<number1> <operator> <number2>`)
- Display result or error from Arduino
- Save the operation history to a .txt file
- Display the date and time in the history saved

Technologies Used

Component	Technology
Programming Language	C++
Compiler	MSVC v143/g++/GCC
Platform	Windows/Linux
Serial Communication	Windows API/termios (Linux)

How It Works

1. User Input: The console app prompts the user to enter a math expression of digits.

2. Data Packaging: The input string is checked and validated and sent to the Arduino via a serial port.
3. Arduino Processing: Arduino receives the expression, performs the operation, and sends the result back to the console app.
4. Result Display: The result is read back by the console app and printed/displayed for the user.
5. Save the expression: When the user is done with the operation and clicks exit, it saves to a result.txt file in the project folder.

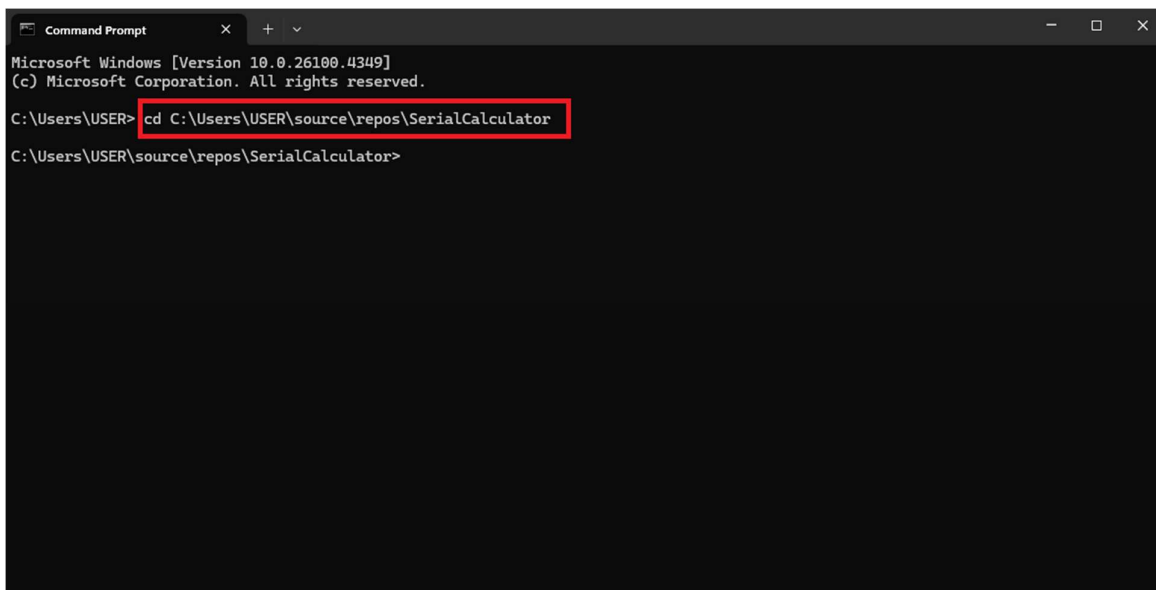
Setup Instructions

A. Prerequisites

- Arduino IDE (Arduino sketch uploaded)
- C++ compiler (g++, Visual Studio, etc)
- Properly connected Arduino (Uno) via USB
- Know the COM port (e.g., COM3 on Windows or /dev/ttyUSB0 on Linux)
- SerialPort-master library (Particularly SerialPort.cpp and SerialPort.hpp in the project folder). See link below.

B. Build Instruction (Using g++)

1. Changing directory to the one intended

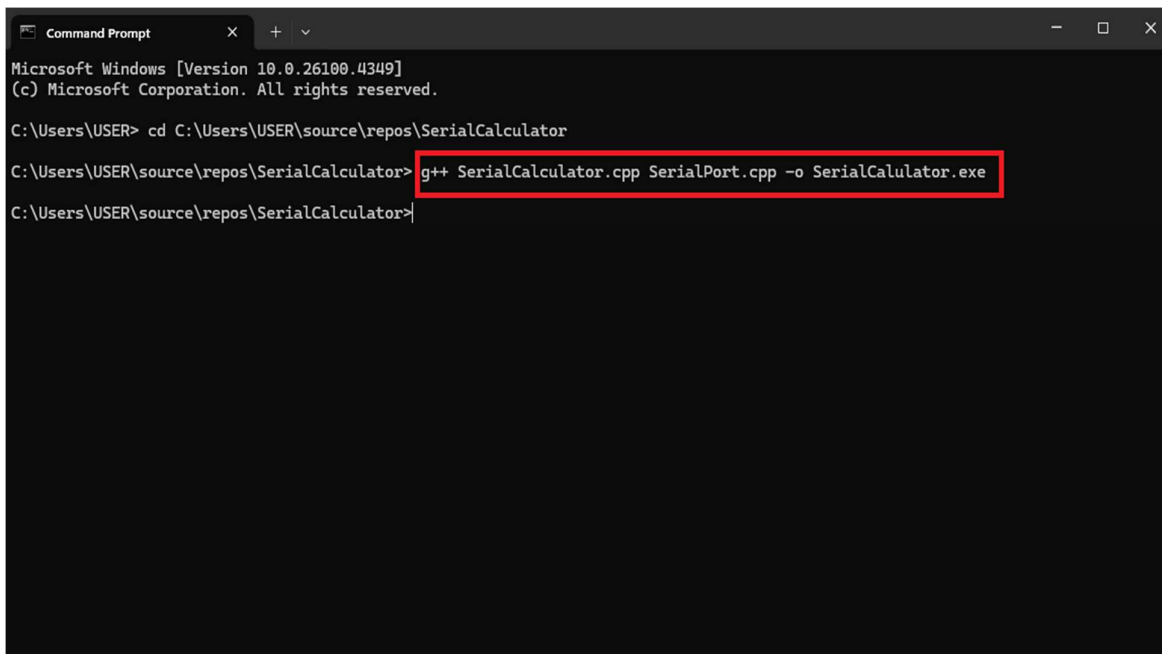


```
Microsoft Windows [Version 10.0.26100.4349]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER> cd C:\Users\USER\source\repos\SerialCalculator
C:\Users\USER\source\repos\SerialCalculator>
```

Figure 2 Changing directory

2. Creating the executable(.exe) file



```
Microsoft Windows [Version 10.0.26100.4349]
(c) Microsoft Corporation. All rights reserved.

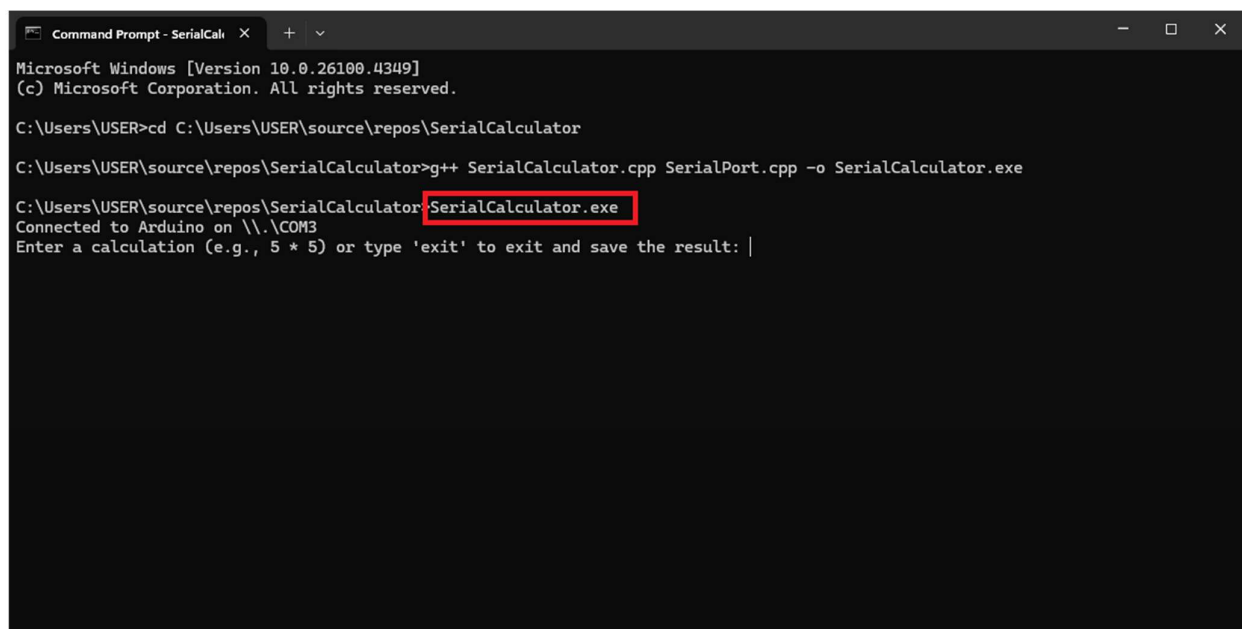
C:\Users\USER> cd C:\Users\USER\source\repos\SerialCalculator

C:\Users\USER\source\repos\SerialCalculator> g++ SerialCalculator.cpp SerialPort.cpp -o SerialCalculator.exe

C:\Users\USER\source\repos\SerialCalculator>
```

Figure 3 Create an executable

C. Running the executable(.exe) file from cmd



```
Microsoft Windows [Version 10.0.26100.4349]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER>cd C:\Users\USER\source\repos\SerialCalculator

C:\Users\USER\source\repos\SerialCalculator>g++ SerialCalculator.cpp SerialPort.cpp -o SerialCalculator.exe

C:\Users\USER\source\repos\SerialCalculator>SerialCalculator.exe
Connected to Arduino on \\.\COM3
Enter a calculation (e.g., 5 * 5) or type 'exit' to exit and save the result: |
```

Figure 4 Running the executable

D. Alternatively, click the .exe file from the project folder



Figure 5 Alt. running the executable

E. Importing SerialPort.cpp

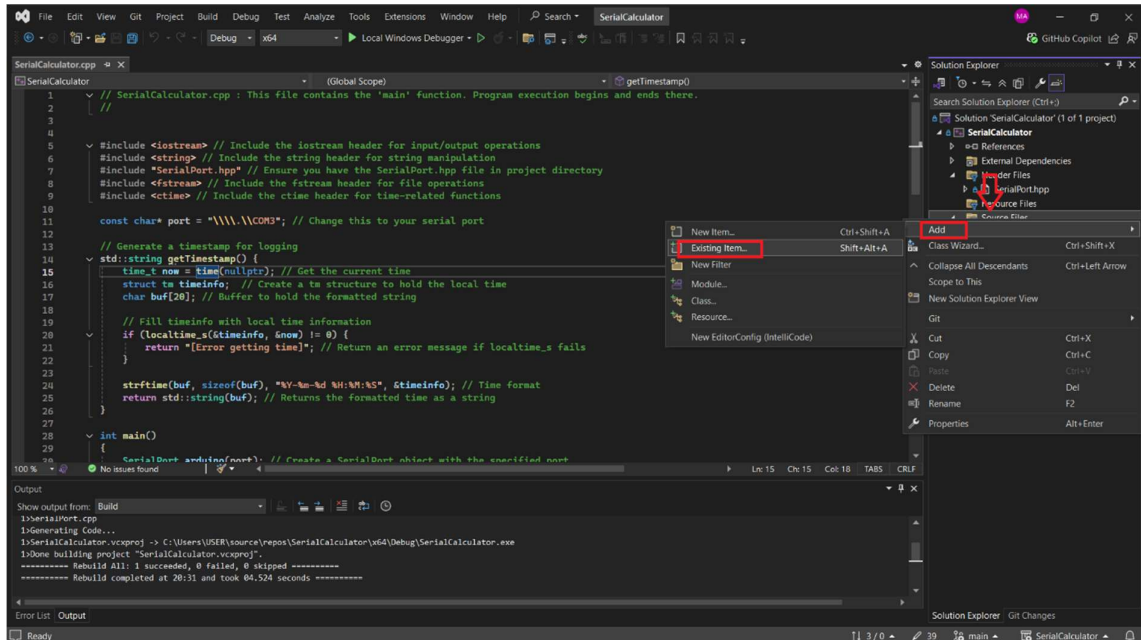


Figure 6 Importing SerialPort.cpp

F. Importing SerialPort.h (header file)

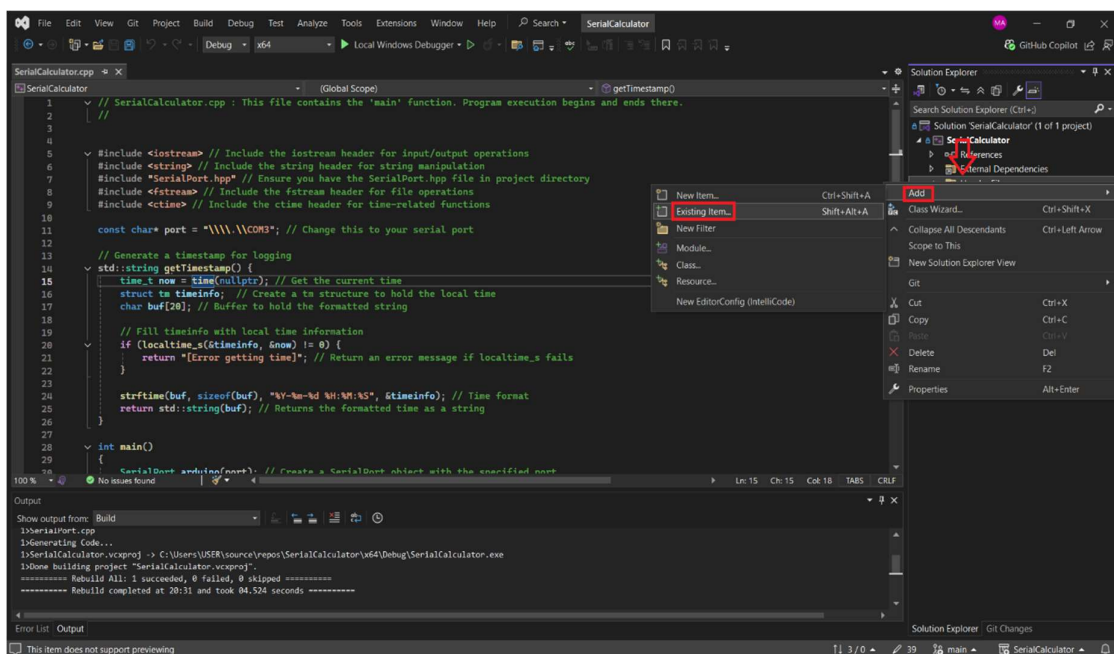


Figure 7 Importing SerialPort.h

G. Build Instructions (Using Visual Studio)

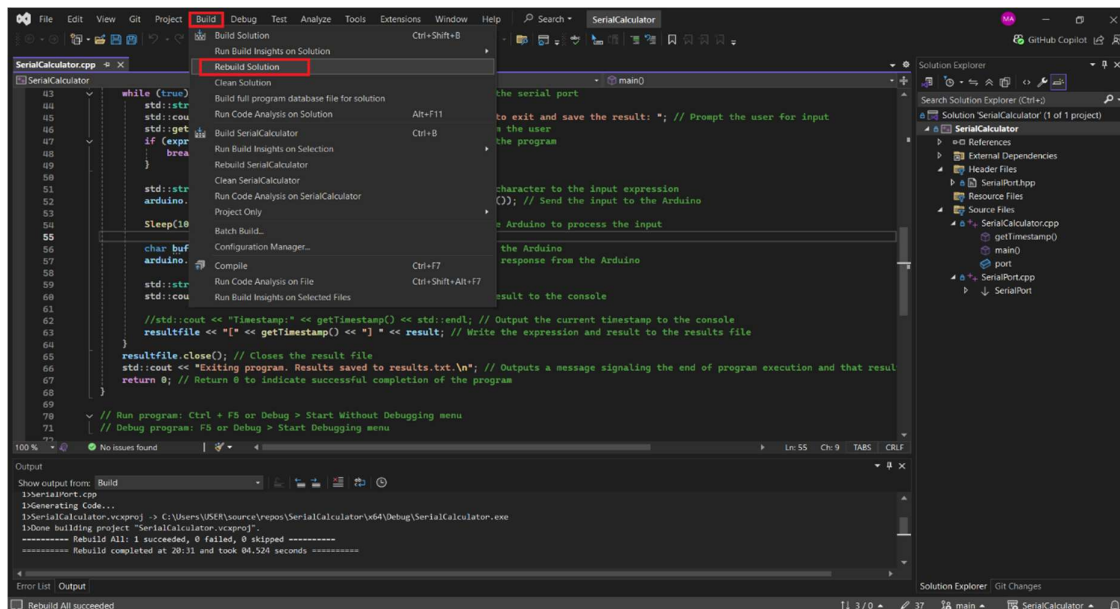


Figure 8 Build on Visual Studio

H. Running the executable(.exe) file from Visual Studio

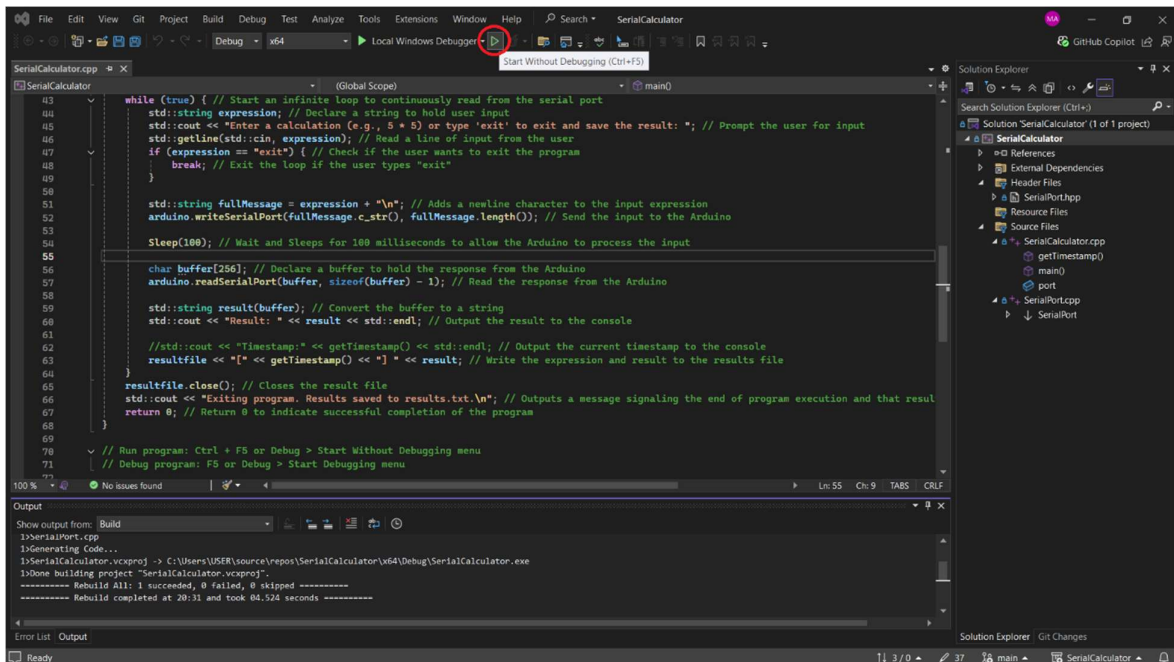


Figure 9 Running executable from Visual Studio

I. Selecting the Arduino board to the microcontroller

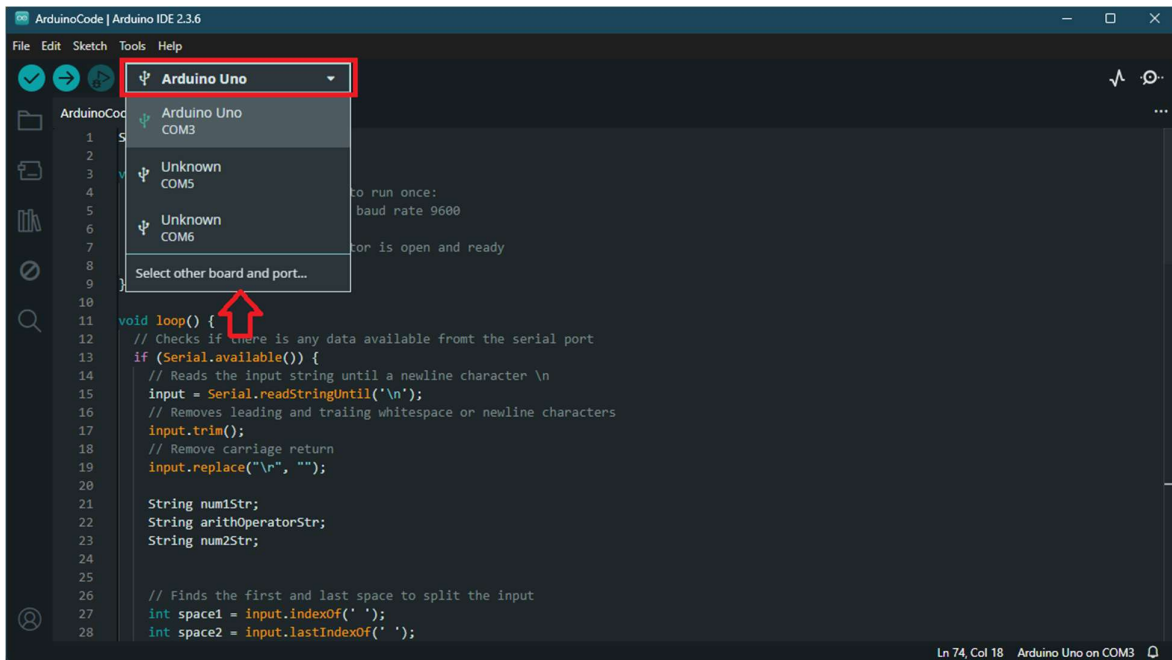


Figure 10 Selecting the board and port

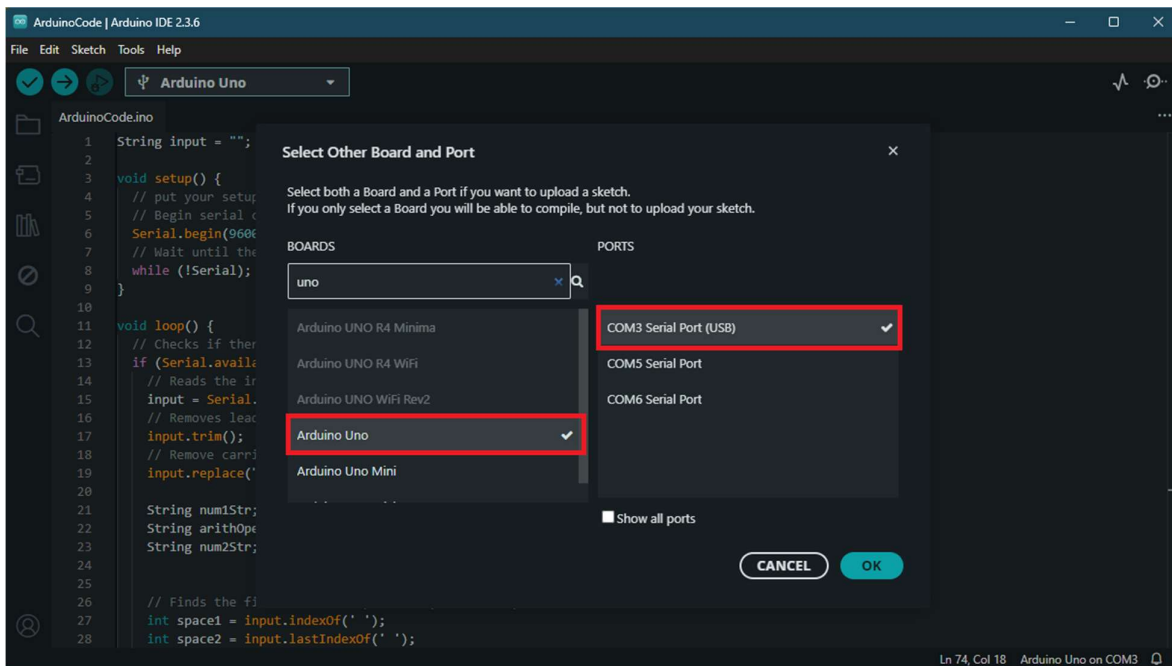


Figure 11 Selecting the board and port extended

J. Uploading the code to the Arduino Uno microcontroller

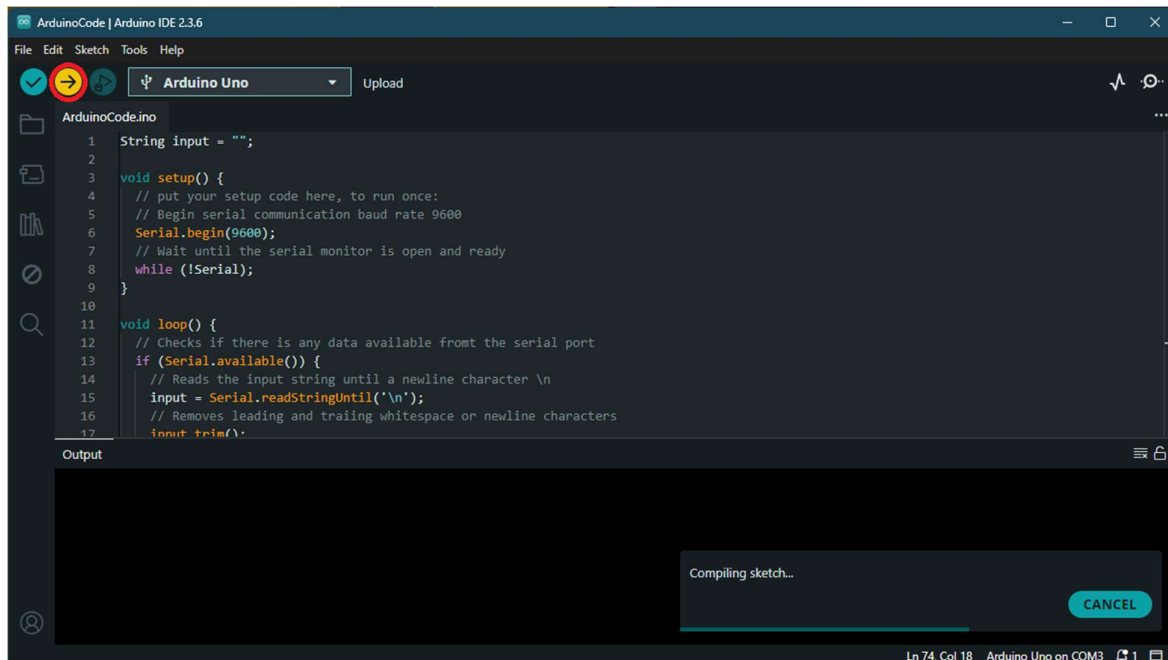


Figure 12 Upload the code to the microcontroller

Example Usage

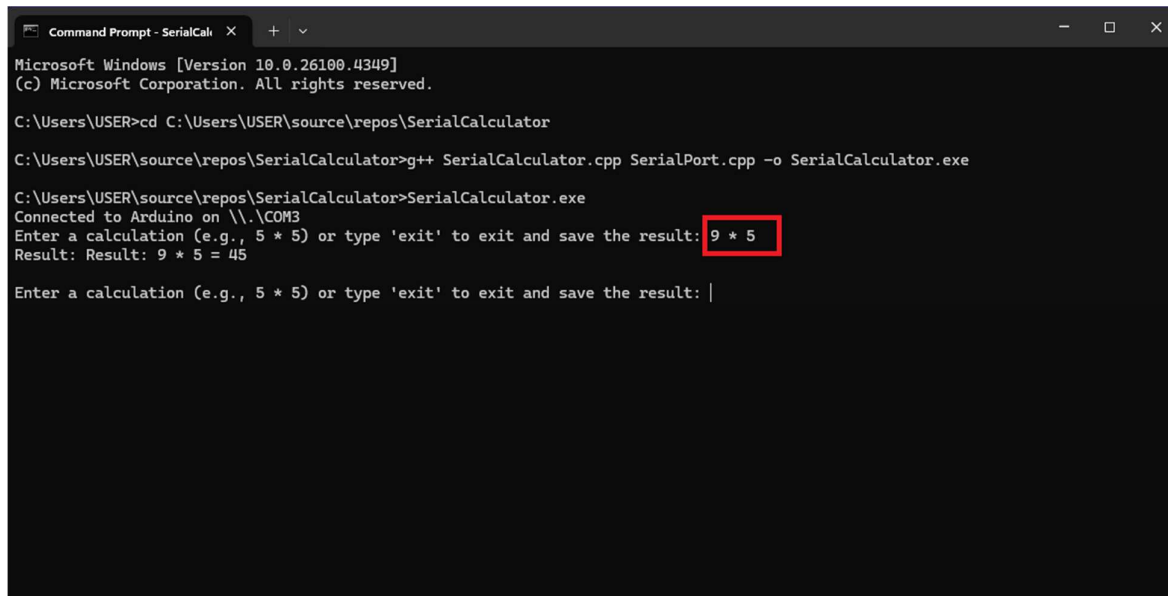


Figure 13 Usage example 1

```
Command Prompt - SerialCali X + -
Microsoft Windows [Version 10.0.26100.4349]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER>cd C:\Users\USER\source\repos\SerialCalculator

C:\Users\USER\source\repos\SerialCalculator>g++ SerialCalculator.cpp SerialPort.cpp -o SerialCalculator.exe

C:\Users\USER\source\repos\SerialCalculator>SerialCalculator.exe
Connected to Arduino on \\.\COM3
Enter a calculation (e.g., 5 * 5) or type 'exit' to exit and save the result: 9 * 5
Result: Result: 9 * 5 = 45

Enter a calculation (e.g., 5 * 5) or type 'exit' to exit and save the result: 9*5
Result: Result: 9 * 5 = 45

Enter a calculation (e.g., 5 * 5) or type 'exit' to exit and save the result: |
```

Figure 14 Usage example 2

Error Handling

1. Arduino not connected → Error message, failed to connect to COM port

```
Command Prompt X + -
Microsoft Windows [Version 10.0.26100.4349]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER> cd C:\Users\USER\source\repos\SerialCalculator

C:\Users\USER\source\repos\SerialCalculator> g++ SerialCalculator.cpp SerialPort.cpp -o SerialCalculator.exe

C:\Users\USER\source\repos\SerialCalculator>SerialCalculator.exe
ERROR: Handle was not attached.Reason : \\.\COM3 not available
Failed to connect to Arduino on\\.\COM3

C:\Users\USER\source\repos\SerialCalculator>|
```

Figure 15 When Arduino is not connected

2. Invalid input format → User prompted to re-enter

```
Command Prompt - SerialCalu x + v
Microsoft Windows [Version 10.0.26100.4349]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER>cd C:\Users\USER\source\repos\SerialCalculator

C:\Users\USER\source\repos\SerialCalculator>SerialCalculator.exe
Connected to Arduino on \\.\COM3
Enter a calculation (e.g., 5 * 5) or type 'exit' to exit and save the result: tt
Result: Error: No valid operator found
Enter a calculation (e.g., 5 * 5) or type 'exit' to exit and save the result: |
```

Figure 16 Invalid input format

3. Division by zero —→ Error message, division by zero

```
Command Prompt - SerialCalu x + v
Microsoft Windows [Version 10.0.26100.4349]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER>cd C:\Users\USER\source\repos\SerialCalculator

C:\Users\USER\source\repos\SerialCalculator>SerialCalculator.exe
Connected to Arduino on \\.\COM3
Enter a calculation (e.g., 5 * 5) or type 'exit' to exit and save the result: 5/0
Result: Error: Division by zero
Enter a calculation (e.g., 5 * 5) or type 'exit' to exit and save the result: |
```

Figure 17 Division by zero error

Testing the Setup

To ensure that the components work as expected:

1. Open the Arduino IDE and open the Serial Monitor (Tools —→ Serial Monitor).
2. Set the baud rate to **9600** or as configured by the user.
3. Type a **mathematical expression** (basic) and press **Enter** to test.

4. If the setup is working correctly, the microcontroller will perform the calculation and display the result.

Note

If the Serial Monitor is open in Arduino IDE and the microcontroller is still connected, the calculator will not be able to connect to the microcontroller via the console app. Hence, the serial port can only be used by one application at a time, either the Arduino IDE or the Console App. So, make sure the Serial Monitor in the Arduino IDE is closed before attempting to use the console.

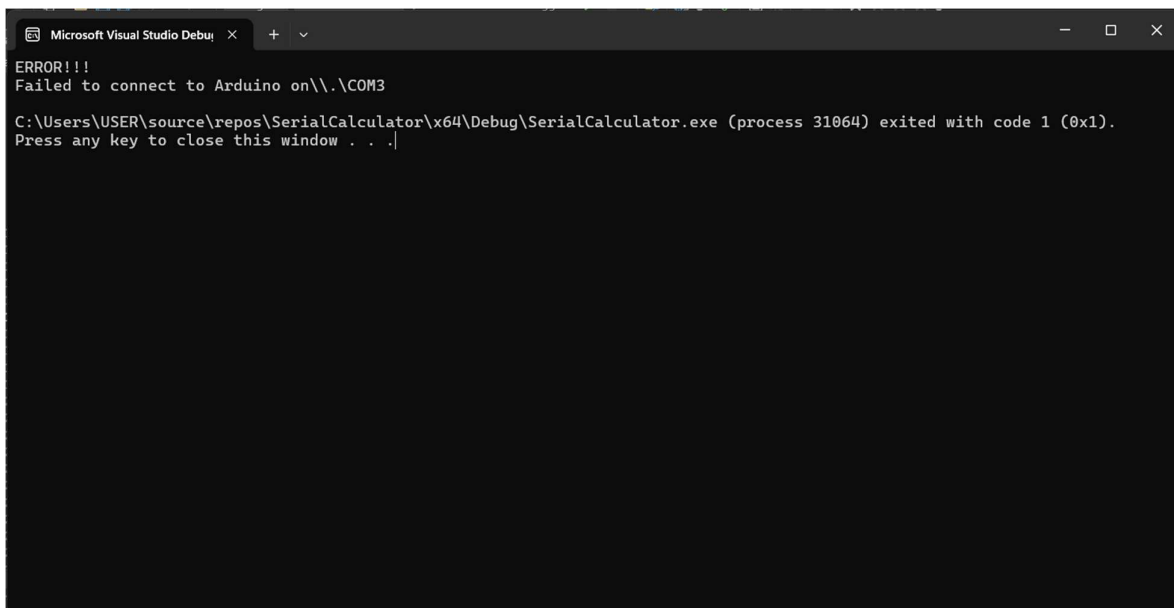


Figure 18 Failed to connect error

Bibliography

Mandal, M (2020, April 21). *SerialPort*. Github. <https://github.com/manashmandal/SerialPort>