**Development Phase of Simple Calculator using Microcontroller**

In this phase we should have the codes ready and communicating normally with each other, we will go into more details about how the structure was built.

First code on the PC side, as previously shown in the first phase, which is responsible for receiving, sending, inputs/outputs, and saving communications, we will obviously need main important functions to be available, many of these function's role described in Microsoft Docs.

We started the code with a User-defined functions, such void logMessage(const std::string& message) that is explained in C++ File Handling at Tutorialspoint, logs the messages into a file named as saved_communication between both codes, we should also mention the int main() documented in cppreference, which is considered as the entry point of the program, coordinating the overall workflow including serial port operations and user interactions. Also in this part we used CreateFile, that creates or opens a file or I/O device, this function is primary function to initialize the connection with the board, with this functions we attempt to open the serial communication port with the Arduino board, the function will try to get a handle to the COM3 channel chosen here, if the channel did not open, we get the error message by the std::cerr function, and the program stops running.

In case we had a communication between the codes, the program proceed with setting the data transmissions and configurations of the communication, GetCommState, retrieves the current control settings for a specified serial port, SetCommState, configures a serial port according to the specifications in a DCB structure, we set here the communication speed (19200), size of data packet (8bits), number of bits (1 stop bits), and set the parity simple error checking method (NOPARITY). In addition to CloseHandle that is used in different chapters of the code in order to close an open object handle and ensures the normal functionality.

Next, the PC code will asks the User to enter the values he wants to calculate and the operation wished, this will be displayed by the std::cin function, these inputs will be stored in a variable defined as (first_number, operation, second_number). The program is also designed to check the if these values are valid, if invalid the, for example the user wrote a letter instead of number, the error message will show (Invalid operation) and the programs stop running. If we got valid operation, what happens next is the std::to_string function will transform these values into string form, for example to (2 , + , 2), in this case, this string is easy to be sent over the serial communication, and as required we also ensured to save it in the saved_communication file, by again the logMessage function.

After that and using WriteFile which writes data to an I/O device, we will send the string to the serial port, the string is converted into bytes that represent specific character, WriteFile function is called with certain parameters, these parameters are, the serial handle which is the handle to the serial port, the bytes array to be written, the number of bytes, the variable to store the number of bytes is actually written. We have no overlapping structure, since it is a simple write operation.

Finally, if hopefully the message is sent, and the operation performed and we are receiving the result, the ReadFile function will in turn read this result by reading the bytes into a buffer, this buffer is a temporary storage area in memory where it will store the data, in our case we determined it as char buffer [512], that mean it can store up to 512 characters, the ReadFile will store the read actual number of bytes in bytesRead. The data being stored in buffer, and then null terminated, this ensure it is valid as a C style string, then it is printed out by the std::cout function to the screen.
We have also supported the program by adding an error function if the message could not been sent and the message will be saved in the saved_communication file and we end by the CloseHandle function.

These functions are the core for the main operations of initializing the communication channel, configuring it, sending and receiving data, and logging the process, making them the crucial components of our PC code, in addition to other functions and components also explained in the comments to complete the functionality of the program.

On the other side, Arduino code uploaded by Arduino Ide, we used couple main functions, which are documented and explained in the Arduino.cc that is the official website of Arduino platform.

At the beginning we start with the void setup(), and, void loop(), together define the main structure of the Arduino program, by initializing settings and running continuously to handle the program's main tasks, we should ensure to have the same baud rate by the Serial.begin() function, which also shows the successful operations by setting the LED built-in pin as an output. The Arduino code next will ensure continuous checking for any available data from PC, it will keep checking if there are any bytes to read from the serial port using the Serial.available() which will also give an infinite loop using the void loop().

Then when we receive the User data in the form of string as explained previously, this string will be read by Serial.readStringUntil() function, this function in turns reads all the characters from the serial buffer and stores them in a variable named (String) until it finds a newline character (\n), in simple words as our example before, it will read 2 + 2 and stops when reaches \n character.

This string will be parsed and get the 2 numbers and the operation symbol needed by finding the positions of the commas in the string and using functions like indexOf and substring to separate the components. The extracted components will be converted from strings to numbers by the String.toDouble function, this function parses the string, read the characters and interpret them as numerical value, after that it is ready to perform the calculation in the board as we defined the calculations operations (addition, subtraction,…) and sure we should add the error message again to support the program for any invalid operation.

At the end, if valid operation, we transfer the result number into a string again by the String() function, this function interprets the numerical value and generate a textual representation as a sequence of characters, that represent this value.

Finally we send this string by the Serial.println(), it write each character of the string to the serial buffer, the functions takes the string and break it into individual characters, each character is sent sequentially over the serial connection, also we added the digitalWrite and delay functions for blinking the LED.

The mentioned functions respectively, simply structure the program, manage serial communication, and process and convert I/O data.

To run the program, we start to download both the Visual Studio Code from its official website to use it as an option code editor, and we also need to install the Arduino Ide from its website, we must choose the correct version that fits our operating system, and install them.

After successfully installing the Arduino IDE, we launch the program, choose the correct port and board used and connect the Arduino board to our computer. Then we open New Sketch from Files tab, and paste the Arduino code. We can directly upload it by clicking on the right arrow button in the top left corner of the program. You should see a message in the console when successfully uploading the code.
For VS code, we must install a compiler, in my case I chosen the G++ compiler, it transforms our source code into executable programs, in addition, we will need to further add the G++ compiler to the system path in the environment variable that will allow our operating system to recognize and execute it in our prompt command.

Next we need to create New Folder to put the code inside it, we can open the folder from VS code, create New File and give it name.cpp, this will create the file inside the folder, you can paste now the code in this file. After that go to the Folder containing the code file, and from up in the path, launch the prompt commander window by typing cmd then click Enter, this will open the commander window, to execute the code using the G++ compiler, type 'g++-o name.exe name.cpp'. Proceed in VS code and open New terminal to run the code, a new window will open down, you can type the name of the code.exe and back slash, this will run the code and should at first produce ' Enter First Number '.

Now you can type the numbers and next the operation to perform, to test the program functionality, try to type high volume of numbers and rapidly, this can show how the program can handle the operations. Also, another way is to ask other users to try it themselves, this also can figure any unexpected problem. In addition,  it is good to try the different kinds of operations, like dividing by zero, typing letters instead of numbers, trying both negative and positive values operations.

Lastly you can sure monitor the program from the serial monitor in Arduino Ide and also the saved_communications file in the VS code, that will save the results performed in the program.

Link for both codes: https://github.com/mohamadk911/Mohamad-Mustapha

Mohamad El Mustapha

Dresden 26 June 2024