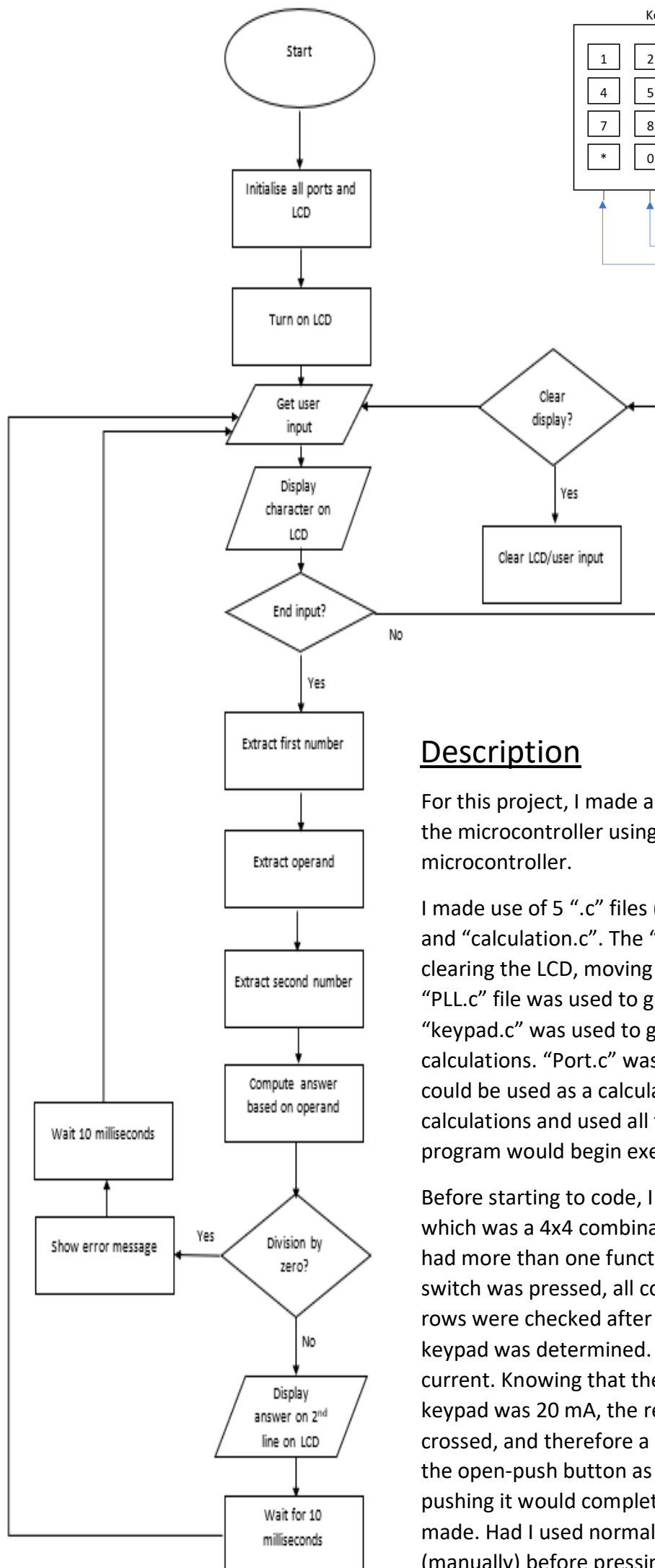
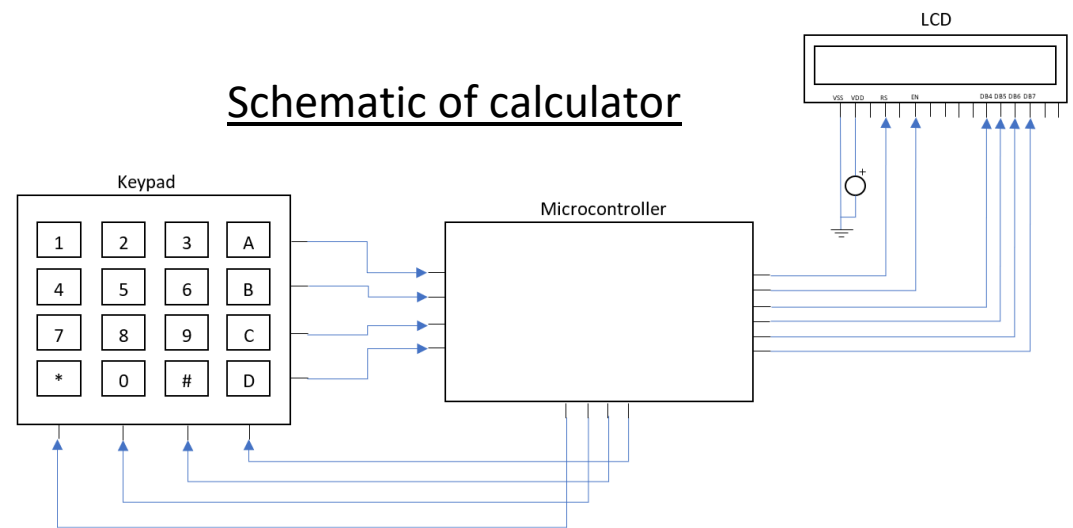


Calculator

Flowchart of calculator



Schematic of calculator



Achievements

- Perform simple calculations (+, -, /, *) of two numbers
- Interface with the LCD (initialise it and display the answers)
- Use a keypad to get the correct values of the corresponding button.
- Successfully extract numbers and operands from a string
- Calculations can be done on integers, floating points, and exponentials (only positive exponentials considered as calculator only shows 2dp, small numbers may not be shown).
- Graphics are displayed on the LCD when division by zero occurs.
- Successfully implemented other functionality like deleting a character, clearing the display etc.

Description






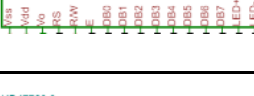
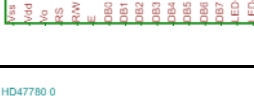


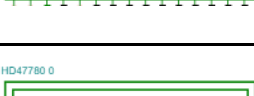
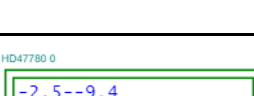
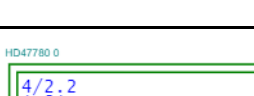
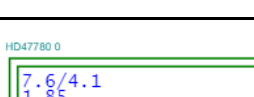
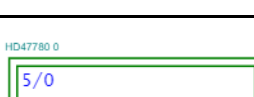
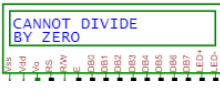
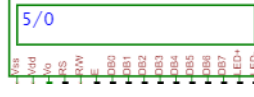

For this project, I made a simple calculator using the TM4C123GH6PM microcontroller. I coded the microcontroller using C in Code Composer Studio and used TinaCloud to simulate the microcontroller.

I made use of 5 ".c" files (along with their header files): "lcd.c", "PLL.c", "keypad.c", "Port.c" and "calculation.c". The "lcd.c" file was used to interface with the LCD, and had commands like clearing the LCD, moving to the next line in the LCD, displaying characters on the LCD etc. The "PLL.c" file was used to generate the clock, which was used when interfacing with the LCD. "keypad.c" was used to get the character pressed by the user, and this was used in the calculations. "Port.c" was used to initialise and set up all the ports before the microcontroller could be used as a calculator. "Calculation.c" was the file responsible for making all the calculations and used all the other files to do so. It also had the main function, where the program would begin execution.

Before starting to code, I designed the circuit on TinaCloud. I started by making the keypad, which was a 4x4 combination of switches, each serving its own purpose. Some switches also had more than one function, which was invoked by pressing the shift key. To know which switch was pressed, all columns of the keypad were made high (1), and then low (0) in turn. The rows were checked after each low, and depending on which row was low, the output of the keypad was determined. The resistors were included in the design to limit the amount of current. Knowing that the microcontroller operated with 3.3V and the operating current for the keypad was 20 mA, the resistor value should be chosen to ensure the 20 mA limit is not crossed, and therefore a 1k resistor did the job ($3.3/1000 = 3.3\text{mA}$). Additionally, I chose to use the open-push button as my switch as it mimicked the traditional button on a calculator; pushing it would complete the circuit only for a few milliseconds, enough for a reading to be made. Had I used normal switches, I would have to close the switch, then open it again (manually) before pressing the next switch.

Next, I set up the LCD, connecting the correct bits of the LCD to the corresponding pins on the microcontroller. Although setting up the HD44780 LCD took a bit longer (as the LCD did not behave the same way in TinaCloud as was described in its datasheet), I managed to initialise it (albeit with some of the delays commented out). I used the LCD in 4-bit mode (DB4- DB7), meaning to send a character, two calls were made, each containing a nibble. To display the digits on the LCD, I only had to send the corresponding byte to the LCD, which was available in the HD44780 data sheet. Additionally, all commands were also available in the datasheet, meaning all I had to do was send the correct byte as an instruction. Finally, since I was always writing to the LCD (never reading from it), the R/W was grounded.

Testing

Test number	Input	Expected output	Actual output	Image of LCD
1	5+5	10	10.00	
2	0+7.8	7.8	7.79	
3	9.9+0.7	10.6	10.59	
4	4.2E3+3.4	4203.4	4203.39	
5	-7.6+8.5	0.9	0.90	
6	8.5+-2.3	6.2	6.20	
7	8-6	2	2.00	
8	0-7.8	-7.8	-7.79	
9	9.9-4.6	5.3	5.30	
10	3.8E2-22	358	358.00	
11	-7.5-9.3	-16.8	-16.79	
12	-2.5--9.4	6.9	6.90	
13	4/2.2	1.81	1.81	
14	7.6/4.1	1.85	1.85	
15	5/0	Error		
16	6E3/40	150	149.99	

17	-6.6/5.9	-1.11	-1.11	
18	5*8	40	40.00	
19	9.2*6	55.2	55.19	
20	3.3*4.2	13.86	13.85	
21	0*7.8	0	0.00	
22	5.8E4*0.5	29000	28999.99	
23	-7.5*9	-67.5	-67.49	