

**s**

**LEROTHOLI POLYTECHNIC**

**SCHOOL OF ENGINEERING**

**AND**

**TECHNOLOGY**

|  |  |
| --- | --- |
| **Student Number:** | **202301320CE** |

|  |  |
| --- | --- |
| **Surname & Other Names:** | **TLOKOTSI POTLOANE** |

|  |  |
| --- | --- |
| **Program:** | **B.ENG.TECH COMPUTER ENGINEERING** |

|  |  |
| --- | --- |
| **Subject Name:** | **Microcontroller Systems 1** |

|  |  |
| --- | --- |
| **Subject Code:** | **MCSY22107** |

|  |  |
| --- | --- |
| **Assignment Number:** | **4** |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Due Date:** | **1** | **5** | **0** | **4** | **2** | **0** | **2** | **4** |

|  |  |
| --- | --- |
| **Lecturer:** | **Mr. T.P Raliete** |

**Declaration of own work**

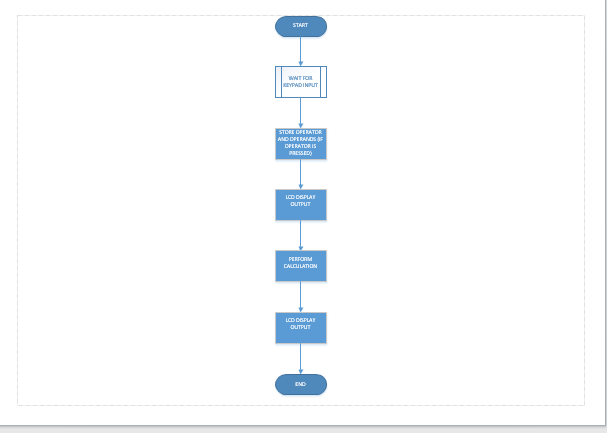
I hereby declare that this assignment is my own work and that it has not been copied from any other person or document.

...T.Potloane.............................. ....15/04/24........................ signature date

**TRUTH TABLE**

|  |  |  |  |
| --- | --- | --- | --- |
| Operand 1 | Operator | Operand 2 | Result |
| 00-99 | + | 00-99 | 00-99 |
| 00-99 | - | 00-99 | 00-99 |
| 00-99 | \* | 00-99 | 00-99 |
| 00-99 | / | 00-99 | 00-99 |
|  |  |  |  |

**FLOW CHART**



**THE CODE**

unsigned short kp;

char oldstate = 0;

int cnt = 0;

char txt[13] = {'\0','\0','\0','\0','\0','\0','\0','\0','\0','\0','\0','\0','\0'};

// Keypad module connections

char keypadPort at PORTD;

// End Keypad module connections

// LCD module connections

sbit LCD\_RS at RB4\_bit;

sbit LCD\_EN at RB5\_bit;

sbit LCD\_D4 at RB0\_bit;

sbit LCD\_D5 at RB1\_bit;

sbit LCD\_D6 at RB2\_bit;

sbit LCD\_D7 at RB3\_bit;

sbit LCD\_RS\_Direction at TRISB4\_bit;

sbit LCD\_EN\_Direction at TRISB5\_bit;

sbit LCD\_D4\_Direction at TRISB0\_bit;

sbit LCD\_D5\_Direction at TRISB1\_bit;

sbit LCD\_D6\_Direction at TRISB2\_bit;

sbit LCD\_D7\_Direction at TRISB3\_bit;

// End LCD module connections

int keyPad() {

// Wait for key to be pressed and released

do {

kp = Keypad\_Key\_Click(); // Store key code in kp variable

} while (!kp);

// Prepare value for output, transform key to its ASCII value

switch (kp) {

case 1: return 49; break; // 1

case 2: return 50; break; // 2

case 3: return 51; break; // 3

case 4: return 65; break; // A

case 5: return 52; break; // 4

case 6: return 53; break; // 5

case 7: return 54; break; // 6

case 8: return 66; break; // B

case 9: return 55; break; // 7

case 10: return 56; break; // 8

case 11: return 57; break; // 9

case 12: return 67; break; // C

case 13: return 42; break; // \*

case 14: return 48; break; // 0

case 15: return 35; break; // #

case 16: return 68; break; // D

}

}

char key;

char input[4] = {0}; // Maximum 3 digits for input plus '\0'

int num1 = 0, num2 = 0, result = 0;

char operatorr = 0;

int input\_length = 0;

void main() {

ANSEL = 0; // Configure AN pins as digital I/O

ANSELH = 0;

Keypad\_Init();

LCD\_Init();

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

//LCD\_OUT(1,1,"Calculator");

delay\_ms(500);

while (1) {

key = Keypad\_Key\_Click();

if (key) {

if (key >= '0' && key <= '9') {

// Append digit to input string

if (input\_length < 3) {

input[input\_length++] = key;

LCD\_Chr(1, input\_length, key);

}

} else if (key == '+' || key == '-' || key == '\*' || key == '/') {

// Store first number and operator

int i;

for (i = 0; i < input\_length; i++) {

num1 = num1 \* 10 + (input[i] - '0');

}

operatorr = key;

LCD\_Chr(2, 1, key);

LCD\_Chr(2, 2, ' ');

LCD\_Chr(2, 3, ' ');

LCD\_Chr(2, 4, ' ');

input\_length = 0; // Clear input

} else if (key == '=') {

// Perform calculation

// Convert the hundreds digit (if present)

if (input[0] >= '0' && input[0] <= '9') {

num2 += (input[0] - '0') \* 100;

}

// Convert the tens digit (if present)

if (input[1] >= '0' && input[1] <= '9') {

num2 += (input[1] - '0') \* 10;

}

// Convert the units digit

if (input[2] >= '0' && input[2] <= '9') {

num2 += (input[2] - '0');

}

LCD\_Chr(2, 4, '=');

switch (operatorr) {

case '+':

result = num1 + num2;

break;

case '-':

result = num1 - num2;

break;

case '\*':

result = num1 \* num2;

break;

case '/':

result = (num2 != 0) ? num1 / num2 : 0; // Handle division by zero

break;

}

// Extract individual digits from result and store them in the input array

input[0] = '0' + (result / 100); // Hundreds digit

input[1] = '0' + ((result / 10) % 10); // Tens digit

input[2] = '0' + (result % 10); // Units digit

input[3] = '\0'; // Null-terminate the input array

LCD\_Out(2, 6, input);

} else if (key == 'C') {

// Clear input and reset variables

input[0] = input[1] = input[2] = input[3] = '\0'; // Clear all elements of the input array

// Clear the contents of the input array

num1 = num2 = result = 0;

operatorr = 0;

LCD\_Cmd(\_LCD\_CLEAR);

}

}

}

}

**SIMULATION CIRCUIT**

