Khalifa University of Science, Technology and Research
Electronic Engineering Department

ELCE333Microprocessor Systems laboratory

Laboratory Experiment 5

SERIAL COMMUNICATION INTERFACE

Lab Partners
Dana Bazazeh, 100038654
Moza Al Ali 100038604

Date Experiment Performed: 25/02/2015
Date Lab Report Submitted: 04/03/2015

Lab Instructor: Dr.Mohamed Al Zaabi/ Dr.MahmoudKhonji

SPRING 2015

Table of Contents

List of Figures and Tables

Summary

In this lab report it is clearly shown using screenshots and figures that students have completely achieved the required tasks and fulfilled the main objectives of this session. This lab contains four tasks and each of these require a different skill. In the first tasks students where asked to print an output using the tera team. However, the second task was just a modification to the first one but with differences in type of output. Apart from that, the focus of the third task was on writing a code using the the ASCII table to allow a user to enter a character and display the selected character on display using the tera team. Finally, the last task was about using the Dip Switches as inputs and the tera team as output and input at the same time.

1. Introduction

A serial communications interface (SCI) is a device that enables the serial (one bit at a time) exchange of data between a microprocessor and peripherals such as printers, external drives, scanners, or mice. In this respect, it is similar to a serial peripheral interface (SPI). But in addition, the SCI enables serial communications with another microprocessor or with an external network. The term SCI was coined by Motorola in the 1970s. In some applications it is known as a universal asynchronous receiver/transmitter (UART).

The SCI contains a parallel-to-serial converter that serves as a data transmitter, and a serial-to-parallel converter that serves as a data receiver. The two devices are clocked separately, and use independent enable and interrupt signals. The SCI operates in a nonreturn-to-zero (NRZ) format, and can function in half-duplex mode (using only the receiver or only the transmitter) or in full duplex (using the receiver and the transmitter simultaneously). The data speed is programmable.

Serial interfaces have certain advantages over parallel interfaces. The most significant advantage is simpler wiring. In addition, serial interface cables can be longer than parallel interface cables, because there is much less interaction (crosstalk) among the conductors in the cable.

The term SCI is sometimes used in reference to a serial port. This is a connector found on most personal computers, and is intended for use with serial peripheral devices.

1.1 Aim

To introduce the students to the programming and the use of the asynchronous serial communication interface in HCS12 microcontroller.

1.2 Objectives

On completion of this experiment the student should be able to:

1- Understand serial I/O and its parameters

2- Write an HCS12 program to send and receive data from and to a PC.

3- To compile, download and debug/test a C program using CodeWarrior C compiler and Dragon12 Plus Trainer board.

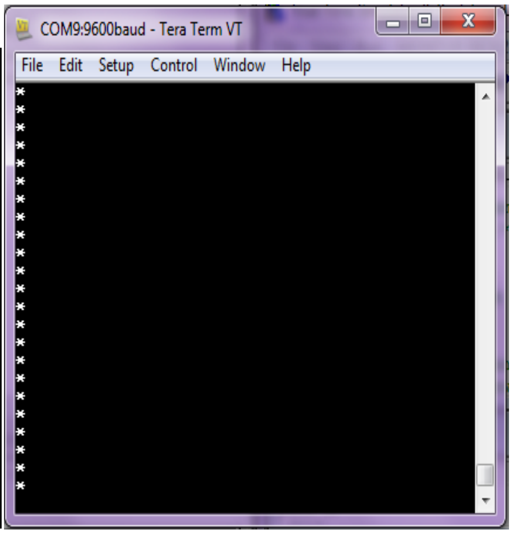
2. Design and Results

Task 1: Transmit Characters

This task asks us to run the code given in the lab script and observe the output in the Tera Term display. The code is shown below in figure 1.

```
#include <hidef.h> /* common defines and macros
*/
#include "derivative.h" /* derivative-specific
definitions */
#include "sci1.h"

void main(void) {
/* put your own code here */
SCI1_Init(BAUD_9600);  //connection
for(;;) {
SCI1_OutChar ('*');// transmits * to SCI1
SCI1_OutChar (0x0A);// new line
SCI1_OutChar (0x0D);// carriage return
} /* loop forever */
/* please make sure that you never leave main */
```

Figure 1: C program for Task1


COM9:9600baud - Tera Term VT
File Edit Setup Control Window Help

```
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
```

Figure 2: Terminal output result

First, we include the header file “sci1.h” that contains functions to access the serial communication interface. Next, we initialize the serial port SCI1 with baud rate in bits/sec using the SCI1_Init(BAUD_9600) function. The function SCI1_OutChar ('*') will print the asterisk character and then we use SCI1_OutChar (0x0A) to create a new line so that each asterisk is on one line. Finally, the SCI1_OutChar (0x0D) function is for carriage return; it will ensure that the printing is done on the beginning of the new line. We make a connection in Tera Term with the appropriate PORT COM number and then run the program on code warrior to observe the results on the display output windowof Tera Turn. Figure 2 above shows the result obtained.

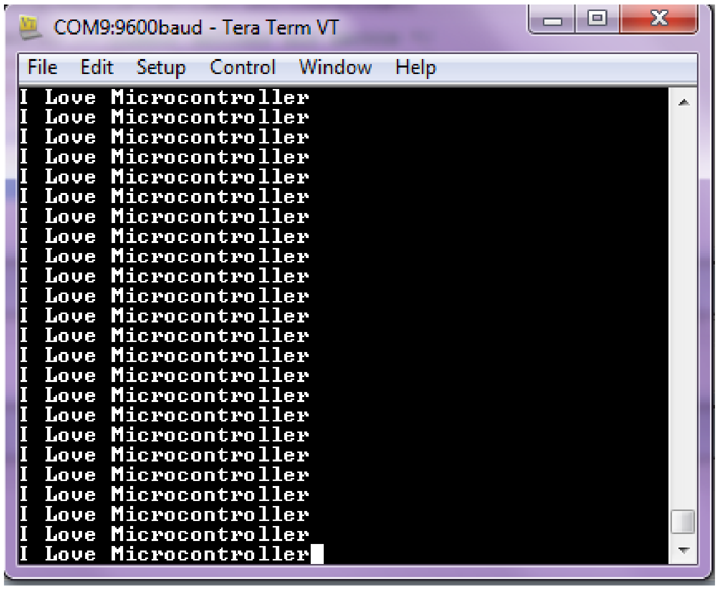
Task 2: Transmit String

This task asks us to modify the program in task 1 to transmit the string “I love Microcontrollers” continuously. Each string transmission should be displayed on the terminal on the start of a new line. Using the table of functions provided in the lab script, all we do is replace the function the prints a character with one that prints a string. This function is SCI1_OutString(“I Love Microcontrollers”). The modified program is shown below in figure 3.

```
#include <hidef.h> /* common defines and macros */
#include "derivative.h" /* derivative-specific definitions */
#include "sci1.h"

void main(void) {
 /* put your own code here */
 SCI1_Init(BAUD_9600);

for(;;) {
 SCI1_OutString ("I Love Microcontroller"); // transmits the sentence to SCI1
 SCI1_OutChar (0x0A);  // new line
 SCI1_OutChar (0x0D); // carriage return
 } /* loop forever */
 /* please make sure that you never leave main */
}
```

Figure 3: C program for Task2

The Tera Term output display is shown below in figure 4.


COM9:9600baud - Tera Term VT
File Edit Setup Control Window Help

```
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
I Love Microcontroller
```

Figure 4: Terminal result for Task 2

Task 3: ASCII Data and Serial I/O

This task asks us to create a program that prompts the user to enter one character, take this character as input, display the ASCII value of this character and then repeat the process again. In addition, the program should display the message “Bye” and terminate when the escape button is pressed. The program output should look something like:

The ASCII code for A is 65

The ASCII code for

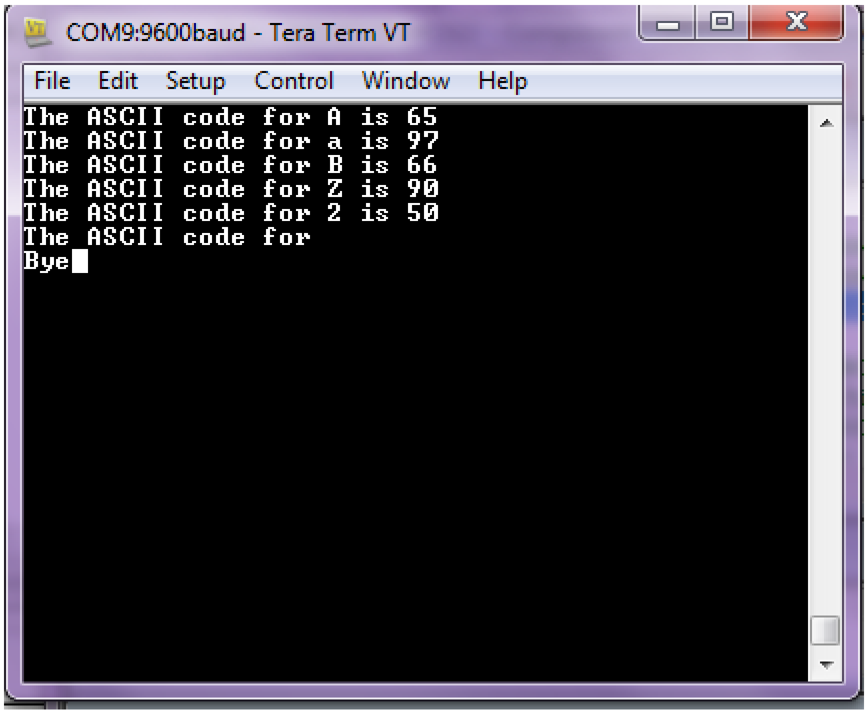
The full program code is shown below in figure 5.

```
#include <hidef.h> /* common defines and macros */
#include "derivative.h" /* derivative-specific definitions */
#include "sci1.h"
void main(void) {
char  x;

/* put your own code here */
SCI1_Init(BAUD_9600);
for(;;) {
SCI1_OutString("The ASCII code for ");// transmits * to SCI1
 x = SCI1_InChar();

if ( x== 27) {
 SCI1_OutChar (0x0A);// new line
SCI1_OutChar (0x0D);
 SCI1_OutString("Bye");
break;
 } else{

 SCI1_OutChar(x);
SCI1_OutString(" is ");// transmits * to SCI1
 SCI1_OutUDec(x);
 }
SCI1_OutChar (0x0A);// new line
SCI1_OutChar (0x0D);// carriage return
} /* loop forever */
/* please make sure that you never leave main */
}
```

Figure 3: C program for Task3

We begin by prompting the user to input a character by displaying “The ASCII code for “using the SCI1_OutString(string) function as previously. Then, we use the function SCI1_InChar() that will return the character entered and save this character in a char variable x. Next, using the conditional if statement, we check whether this character is equal to the escape key by comparing it with 27 which is the ASCII value of esc button that we obtained from an ASCII table. If the condition is met, then we create a new line and display the text “Bye: and break the infinite loop to terminate program. Otherwise, we move on and display the character entered using the function SCI1_OutChar(x) and its equivalent ASCII value using the function SCI1_OutUDec(x); which takes the character as a parameter and writes its ASCII equivalent in unsigned decimal format.

We run the program on Code Warrior and on The Tera Term, test several cases of input characters, including thr escape button condition. Figure 6 below shows the result for several cases.


COM9:9600baud - Tera Term VT
File Edit Setup Control Window Help

```
The ASCII code for A is 65
The ASCII code for a is 97
The ASCII code for B is 66
The ASCII code for Z is 90
The ASCII code for 2 is 50
The ASCII code for
Bye
```

Figure 6: Testing several case on terminal for Task 3

Task-4: Dip Switches Calculator

This task asks us to write a C program that reads 2 4-bit numbers from the DIP switches, the first number taken from the lower 4bits and the second number from the higher 4 bits. These numbers should then be printed on the output terminal and the user should be prompted to choose 1 of the 4 arithmetic operations (+, -, /, *). The operation chosen is then carried out and the result displayed on the terminal. A sample program run is shown below:

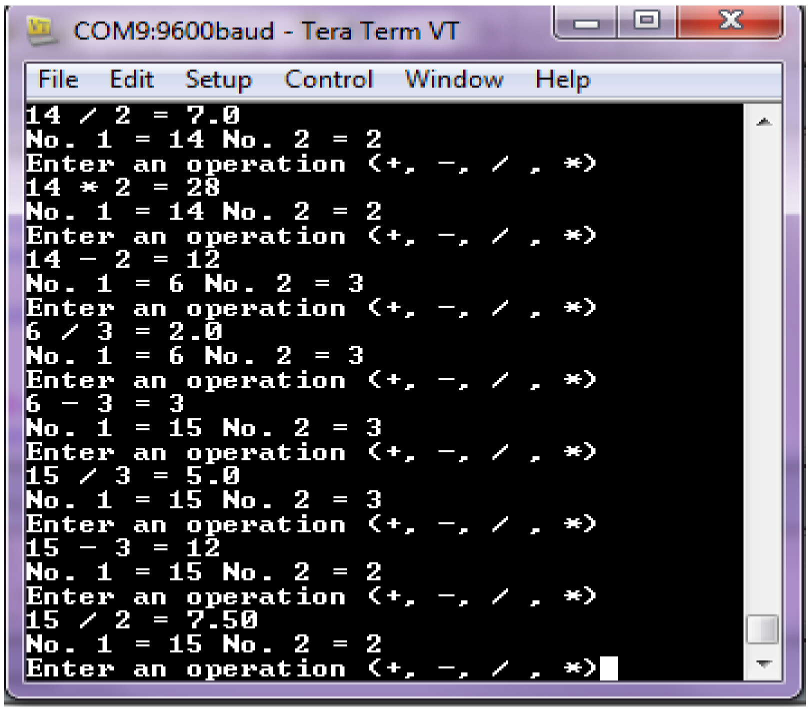
```
No.1= 8 No.2= 15
Enter an operation (+, -, /, *)
8+15=23
```

The full program code is shown below in figure 7.

```
#include <hidef.h>
#include "derivative.h"
#include "sci1.h"

void main(void)
{short num1;
short num2;
short x;
float z;
          SCI1_Init(BAUD_9600);
             EnableInterrupts;
   DDRH = 0x00 ;//set PTH as input

for(;;)
  {
num1 =  ((PTH &0xF0 )>> 4);
num2 = PTH & 0x0F;

   SCI1_OutString("No. 1 = ");
   SCI1_OutUDec(num1);
   SCI1_OutString(" No. 2 = ");
   SCI1_OutUDec(num2);
   SCI1_OutChar (0x0A);// new line
   SCI1_OutChar (0x0D);
   SCI1_OutString("Enter an operation (+, -, / , *)");

    x = SCI1_InChar(); //take in operation char
    SCI1_OutChar (0x0A);// new line
    SCI1_OutChar (0x0D);

if(x==43) { //check if addition
   SCI1_OutUDec(num1);
    SCI1_OutString(" + ");
    SCI1_OutUDec(num2);
    SCI1_OutString(" = ");
    SCI1_OutUDec(num1+num2);
    SCI1_OutChar (0x0A);// new line
    SCI1_OutChar (0x0D);
    }
else if(x==45){ //check if subtraction
     SCI1_OutUDec(num1);
    SCI1_OutString(" - ");
    SCI1_OutUDec(num2);
    SCI1_OutString(" = ");
if(num2>num1) {//negative case
    SCI1_OutString(" - ");
    SCI1_OutUDec(num2-num1);
    } else{

     SCI1_OutUDec(num1-num2);
        }
    SCI1_OutChar (0x0A);// new line
    SCI1_OutChar (0x0D);
    }

else if(x==47) { //check if division
    SCI1_OutUDec(num1);
    SCI1_OutString(" / ");
    SCI1_OutUDec(num2);

    SCI1_OutString(" = ");
     z = num1/num2;
    SCI1_OutFloat(z,num2,num1); //float result
    SCI1_OutChar (0x0A);// new line
    SCI1_OutChar (0x0D);
    }
else if(x==42) { //check if multiplication
    SCI1_OutUDec(num1);
    SCI1_OutString(" * ");
    SCI1_OutUDec(num2);
    SCI1_OutString(" = ");
    SCI1_OutUDec(num1*num2);
    SCI1_OutChar (0x0A);// new line
    SCI1_OutChar (0x0D);
    }
  }
}
```

Figure 7: C program for DIP switches calculator

We begin by setting the data direction register of PTH to input and then reading the 2 numbers using bit masking. The 4 higher bits are obtained by ANDing PTH with 0XF0 and shifting the result 4 bits to the right while the 4 lower bits are obtained directly by ANDing PTH with 0x0F. We save these 2 numbers using 2 variables of type short since this is the type taken as a parameter in the function SCI1_OutUDec(unsigned short). We display these 2 numbers on the same line and then on the next line we prompt the user to choose an operation. This is done in a way similar to task 3, by comparing the ASCII value of the input character entered with the ones for each operation. The ASCII values for +, -, / and * are 43, 45, 47 and 42 respectively. Therefore, using if else statement, we check each case and carry out the selected operation. The division operation should result in a float and therefore we define a float variable and use the function SCI1_OutFloat(z,num2,num1) where z is the division result. In addition, the subtraction operation should consider the case where num2 is greater than num1 and display a negative value. This is done by inserting a nested if statement checking if num2> num1 in which we do num2-num1 instead of num1-num2 and display the negative sign character prior to outputting the result. Figure 8 below shows a sample run with several operations selected.


COM9:9600baud - Tera Term VT
File Edit Setup Control Window Help

```
14 / 2 = 7.0
No. 1 = 14 No. 2 = 2
Enter an operation (+, -, / , *)
14 * 2 = 28
No. 1 = 14 No. 2 = 2
Enter an operation (+, -, / , *)
14 - 2 = 12
No. 1 = 6 No. 2 = 3
Enter an operation (+, -, / , *)
6 / 3 = 2.0
No. 1 = 6 No. 2 = 3
Enter an operation (+, -, / , *)
6 - 3 = 3
No. 1 = 15 No. 2 = 3
Enter an operation (+, -, / , *)
15 / 3 = 5.0
No. 1 = 15 No. 2 = 3
Enter an operation (+, -, / , *)
15 - 3 = 12
No. 1 = 15 No. 2 = 2
Enter an operation (+, -, / , *)
15 / 2 = 7.50
No. 1 = 15 No. 2 = 2
Enter an operation (+, -, / , *)
```

Figure 8: Terminal results for task 4 with several test cases

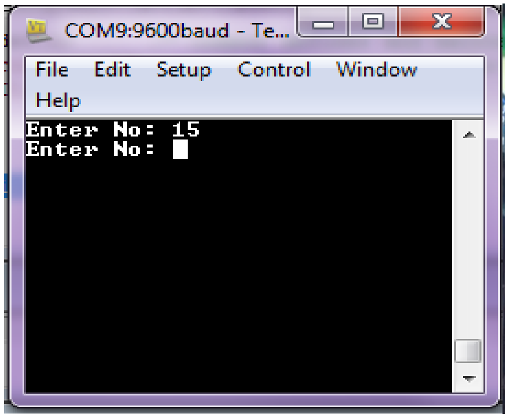
3. Assignment Questions

1) Write an HCS12 program in C to receive bytes serial and put them on PORTB. Set the baud rate at 19200, 8-bit data.

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"      /* derivative-specific definitions */
#include "SCI1.h"

void main(void) {

short x;
    SCI1_Init(BAUD_19200);
    DDRB = 0xFF;    //PORTB as output since LEDs are connected to it
    DDRJ = 0xFF;    //PTJ as output to control Dragon12+ LEDs
    PTJ = 0x0;        //Allow the LEDs to display data on PORTB pins

for(;;) {
  SCI1_OutString("Enter No: ");
x = SCI1_InUDec();
     SCI1_OutChar (0x0A);// new line
    SCI1_OutChar (0x0D);
    PORTB = x;
    }
}
```

Figure 9: C program for Assignment 1

Here we have to first make a connection with PORT B as we did in previous labs. Then, we prompt the user to enter a number using the function SCI1_OutString("Enter No: ") and take in this number in unsigned short decimal and store it in the variable x. Then use PORTB = x; will store this value to PORTB. The results are shown below:



```
Enter No: 15
Enter No: 
```

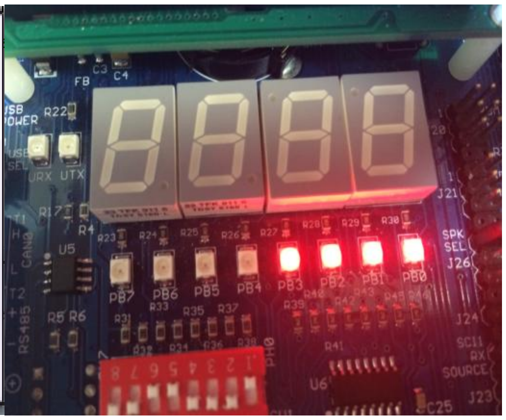


Figure 10: Assignment 1 terminal result **Figure 11: LEDs result when 15 is entered**

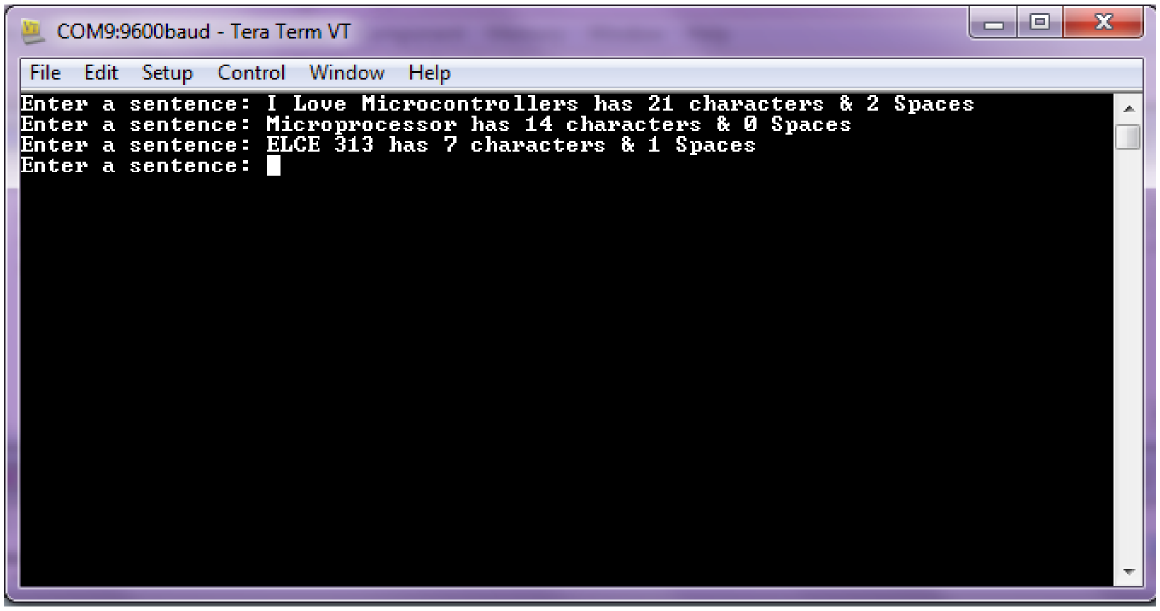
2) Write a program to prompt the user to enter a sentence and then count the number of characters excluding spaces as well as the number of spaces in the entered sentence.

Then display the numbers on the serial terminal. Run the program and provide snap shot of your serial terminal.

```
#include <hidef.h>
#include "derivative.h"
#include "sci1.h"

void main(void) {

            int n=0, cntChar=0, cntSpace=0;
            char array[100];

            SCI1_Init(BAUD_9600);
for(;;)
{   n=0; cntChar = 0; cntSpace = 0;
            SCI1_OutString ("Enter a sentence: ");
SCI1_InString(array, 100);
SCI1_OutString( array);
SCI1_OutString (" has ");

while(array[n]!= 0 ) {
if(array[n] == 32)
cntSpace++;
else
cntChar++;
n++;
}

SCI1_OutUDec(cntChar-1);
SCI1_OutString (" characters & ") ;
SCI1_OutUDec(cntSpace);
SCI1_OutString (" Spaces");
SCI1_OutChar (0x0A);
SCI1_OutChar (0x0D);
}
n=0; cntChar = 0; cntSpace = 0;
 }
```

Figure 12: C program for Assignment 2

We begin by initializing 2 counters, 1 for no. of spaces and the other for characters. We also initialize an index n for the array of characters.We prompt the user to enter a sentence and then use the function SCI1_InString(Array, 100); which will keep reading characters and storing them in the Array until a carriage return is inputtedor until max of 100 characters are entered. Next, we output the entire character array storing the string. Using a while loop, we check if we reached the end of the character array by checking if the character is null = 0. If not, we check whether the next array element is a space by comparing it with the spacebar ASCII value 32. If it is, we increment the spaces counter, otherwise, it is a normal character and we increment the characters counter. Finally after checking all characters, we display the value of both counters. When displaying the character counter, we subtracted 1 from the counter because it adds the enter character, which is not part of the sentence. Figure 13 below shows the terminal output results:


COM9:9600baud - Tera Term VT
File Edit Setup Control Window Help

```
Enter a sentence: I Love Microcontrollers has 21 characters & 2 Spaces
Enter a sentence: Microprocessor has 14 characters & 0 Spaces
Enter a sentence: ELCE 313 has 7 characters & 1 Spaces
Enter a sentence:
```

Figure 13: Assignment 2 terminal result

4. Conclusions

After students have completed of this laboratory session successfully, which by the results they have achieved, analyzing the results and comparing them with the theory we have acknowledged before, we can say that both results match. Moreover, students have learned new techniques and gained general knowledge about the use of serial I/O and its parameters, apart from that they also have learnt how to write a program that communicates with the pc through the board and last but not least how to compile, download, and debug a C program.

In task 1, a concept of printing an output using tera team was introduced. In the second task, a modification had to be done onto the previous program used to allow for a different type of output. In the third task, it is required to write a program that produces a calculator which uses the ASCII table to allow a user to enter a character and display the selected character on display using the Serial Terminal. The final task considered the use of Dip Switches as inputs and the Serial Terminal as output and input at the same time.

Students got maximum benefit from this lab session and many new techniques were introduced to them during the lab session, this widens their knowledge and skills in programming.