

**Khalifa University of Science, Technology and Research**

**Electronic Engineering Department**

**ELCE332 Microprocessor Systems laboratory**

**Laboratory Experiment 5**

**Serial Communication Interface**

**Lab Partners**

Hamad Eisa Alhazami 100036985

Jasim Alhammadi 100035510

**Date Experiment Performed**: 25-2-2015

**Date Lab Report Submitted**: 4-3-2015

**Lab Instructor: Mr. Mohammed Alzaabi/ Mr. Mahmoud Khounji**

**Spring 2015**

# Summary

The purpose of this report is to discuss the results obtained after conducting the four main tasks in the laboratory session. The first task explained how to print out an output on the Tera Term. The second task is a slight modification on the first one to show a different output. The third task was about developing a program that asks the user to input any character; the output, which is the ASCII value of that character, will be displayed on the Tera Term. The fourth task required using the DIP switches and the Serial Terminal as inputs to an arithmetic operation, the output will then be displayed on the Serial Terminal.

Contents

[Summary 2](#_Toc413237400)

[1. Introduction 4](#_Toc413237401)

[1.1 Aim 4](#_Toc413237402)

[1.2 Objectives 4](#_Toc413237403)

[2. Design and Result 5](#_Toc413237404)

[2.1. Task 1: Transmit Characters 5](#_Toc413237405)

[2.2. Task 2: Transmit Strings 6](#_Toc413237406)

[2.3. Task 3: ASCII Data and Serial I/O 7](#_Toc413237407)

[2.4. Task 4: DIP switches Calculator 8](#_Toc413237408)

[3. Analysis and Interpretation 11](#_Toc413237409)

[4. Assignment Questions 12](#_Toc413237410)

[5. Conclusions and Recommendations 14](#_Toc413237411)

# Introduction

SCI or Serial Communication Interface is an interface that is designed for transferring data on asynchronous mode, a mode which utilizes the EIA 232 standard. Asynchronous Serial Data Communication is often used in data communication between data terminal equipment (DTE) which is mainly a computer or a terminal, and data communication equipment (DCE), a modem in most cases. Parallel communication requires a large number of I/O pins to operate; this prevents the microcontroller from connecting with many I/O devices. This could prove to be very costly; hence, SCI is used ahead of parallel communication. Data Communication links come in 3 kinds, simplex, half-duplex and full-duplex.

## 1.1 Aim

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

# Design and Result

## Task 1: Transmit Characters

In the first task, it is required to run the provided program on the Dragon 12 Plus board using Code Warrior and check the output on Tera Term display. The provided program is run on the board. The code for the program is shown below:

**#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\_OutChar ('\*');// transmits \* to SCI1**

**SCI1\_OutChar (0x0A);// new line**

**SCI1\_OutChar (0x0D);// carriage return**

**} /\* loop forever \*/**

**/\* please make sure that you never leave main \*/**

**}**

The provided program will print infinite stars. This can be implemented using a loop and the function “**SCI1\_OutChar ('char')”** to print the character “\*”. Also, the functions **“SCI1\_OutChar (0x0A)”** and **“SCI1\_OutChar (0x0D)”** are used to print the next star in new line. As observed in Tera Term display, infinite lines of stars are printed.

## Task 2: Transmit Strings

In the second task, it is required to modify the provided program to print the string ‘ I love Microprocessors” continuously, run it on the Dragon 12 plus board using Code Warrior, and check the output on Tera Term display. The provided program is modified and run on the board. The code for the program is shown below:

**#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 \* to SCI1**

**SCI1\_OutChar (0x0A);// new line**

**SCI1\_OutChar (0x0D);// carriage return**

**} /\* loop forever \*/**

**/\* please make sure that you never leave main \*/**

**}**

The provided program will print the string “I love Microprocessors” continuously. This can be implemented using a loop and the function “**SCI1\_OutString ("stirng")”** instead of the function “**SCI1\_OutChar ('Char')”** to print the character “\*”. Also, the same functions “**SCI1\_OutChar (0x0A)**” and “**SCI1\_OutChar (0x0D)**” are used to print the next string in new line. As observed in Tera Term display, infinite lines of the previous string are printed.

## Task 3: ASCII Data and Serial I/O

In the third task, it is required to create a program that asks the user to enter a character and display its ASCII code continuously. If the user presses the ESC button, the program should stop and display the message “Bye”. The provided program is written and run on the board. The code for the program is shown below:

**#include <hidef.h> /\* common defines and macros \*/**

**#include "derivative.h" /\* derivative-specific definitions \*/**

**#include "sci1.h"**

**void main(void) {**

**char x;**

**SCI1\_Init(BAUD\_9600);**

**//SCI1\_OutString ("The ASCII code for ");// transmits \* to SCI1**

**for(;;) {**

**SCI1\_OutString ("The ASCII code for ");// transmits \* to SCI1**

**x= SCI1\_InChar ();// input**

**if ( x == 0x1B ) {**

**SCI1\_OutChar (0x0A) ; // new line**

**SCI1\_OutChar (0x0D); // carriage return**

**SCI1\_OutString ("Bye");**

**break;**

**}**

**else {**

**SCI1\_OutChar (x);**

**SCI1\_OutString (" is ");**

**SCI1\_OutUDec (x);**

**SCI1\_OutChar (0x0A);**

**SCI1\_OutChar (0x0D);**

**//break;**

**}}/\* please make sure that you never leave main \*/}**

The program algorithm consists of a loop containing two “if” conditions to check the input character. If the character is **ESC**, the program will display the message “Bye” on Tera Term display and break the loop. Otherwise, it will print the ASCII code for the character. To read the input character, the function “**SCI1\_InChar ()”** is used. Also, the function “**SCI1\_OutUDec (variable)**” is used to print the ASCII code for the character. As observed in Tera Term display, the ASCII value for the provided input character is printed. Besides, when the ESC button is pressed, the message “Bye” is displayed on the screen and the program is stopped.

## Task 4: DIP switches Calculator

In the fourth task, it is required to create a program that reads two numbers from the DIP switches, and the user will enter the operation (+, -, \*, and /) between the two numbers. The program should conduct the arithmetic operation between the two numbers and display the result. The provided program is written and run on the board. The code for the program is shown below:

**#include <hidef.h> /\* common defines and macros \*/**

**#include "derivative.h" /\* derivative-specific definitions \*/**

**#include "sci1.h"**

**void main(void) {**

**char x;**

**int num1;**

**int num2;**

**float z;**

**SCI1\_Init(BAUD\_9600);**

**DDRH= 0x00;**

**for(;;) {**

**num1= PTH\_PTH0\*1 + PTH\_PTH1\*2 + PTH\_PTH2\*4 + PTH\_PTH3\*8;**

**num2= PTH\_PTH4\*1 + PTH\_PTH5\*2 + PTH\_PTH6\*4 + PTH\_PTH7\*8;**

**SCI1\_OutChar (0x0A); // new line**

**SCI1\_OutChar (0x0D); // carriage return**

**SCI1\_OutChar (0x0A); // new line**

**SCI1\_OutChar (0x0D); // carriage return**

**SCI1\_OutString ("num1= ");**

**SCI1\_OutUDec (num1);**

**SCI1\_OutString (", No.2= ");**

**SCI1\_OutUDec (num2);**

**SCI1\_OutChar (0x0A); // new line**

**SCI1\_OutChar (0x0D); // carriage return**

**SCI1\_OutChar (0x0A); // new line**

**SCI1\_OutChar (0x0D); // carriage return**

**SCI1\_OutString ("Enter an operation (+,-,/,\*) ");**

**SCI1\_OutChar (0x0A); // new line**

**SCI1\_OutChar (0x0D); // carriage return**

**SCI1\_OutChar (0x0A); // new line**

**SCI1\_OutChar (0x0D); // carriage return**

**x= SCI1\_InChar();**

**if(x== '+') {**

**SCI1\_OutUDec (num1);**

**SCI1\_OutChar (x);**

**SCI1\_OutUDec (num2);**

**SCI1\_OutChar ('=');**

**SCI1\_OutUDec (num1+num2);**

**} else if(x== '-') {**

**SCI1\_OutUDec (num1);**

**SCI1\_OutChar (x);**

**SCI1\_OutUDec (num2);**

**SCI1\_OutChar ('=');**

**if(num2>num1) {**

**SCI1\_OutChar ('-');**

**SCI1\_OutUDec (num2-num1);**

**} else**

**SCI1\_OutUDec (num1-num2);**

**} else if(x== '\*') {**

**SCI1\_OutUDec (num1);**

**SCI1\_OutChar (x);**

**SCI1\_OutUDec (num2);**

**SCI1\_OutChar ('=');**

**SCI1\_OutUDec (num1\*num2);**

**} else if(x== '/') {**

**SCI1\_OutUDec (num1);**

**SCI1\_OutChar (x);**

**SCI1\_OutUDec (num2);**

**SCI1\_OutChar ('=');**

**z= num1/num2;**

**SCI1\_OutFloat (z,num1, num2);**

**} //else }/\* please make sure that you never leave main \*/}**

The program algorithm consists of a loop containing four “if” conditions to perform the four different operations (+, -, \*, and /). In order to enable and read the DIP switches, the register DDRH value is set to “0x00”. The value of the numbers will be set using the lines “ **num1= PTH\_PTH0\*1 + PTH\_PTH1\*2 + PTH\_PTH2\*4 + PTH\_PTH3\*8;”** and **“num2= PTH\_PTH4\*1 + PTH\_PTH5\*2 + PTH\_PTH6\*4 + PTH\_PTH7\*8;” .** By using this syntax, the value of the two numbers will be directly saved in decimal form in order to perform the operations directly without converting from HEX to decimal. Also, The function “**SCI1\_InChar ()”** is used to read the operation character from the keyboard. As observed in Tera Term display, the values of the two numbers are displayed, as well as the result after conducting the operation.

# Analysis and Interpretation

The four tasks are done using the provided library "sci1.h" in order to use the functions. In the first task, the line code SCI1\_OutChar ('char'); reads 2 nibbles (two HEX values). The same line code is used for printing the new line by using the HEX value (0x0A)of the new line, while the HEX value (0x0D) is used for the carriage return - to insure the new line is printed at the start of the line. The second task is similar to the first one, and the only difference is using the string print function instead of the character print function.

In the third task, in order the check the ESC condition, the “if” condition should check for the HEX value of ESC which is equal to 0x1B. In the fourth task, the value of the numbers is set by defining the left most four DIP switches for the first number and the right most four DIP switches for the second number. Also, the arithmetic operation involving the two numbers is executed using the unsigned representation. Thus, for subtraction, two nested if conditions are used to check for the greater number, then perform the operation.

# 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"**

**intASCII\_Eq(char c);**

**void main(void) {**

**char a=0;**

**SCI1\_Init();**

**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\_OutChar(a);**

**PORTB = ASCII\_Eq(a);**

**}**

**}**

**intASCII\_Eq(char c){**

**return c;**

**}**

**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. Here is what the program would look after entering some test strings**

***Enter a sentence: I Love ELCE has 9 characters & 2 spaces***

***Enter a sentence: Dragon 12 has 8 characters & 1 spaces***

***Enter a sentence:***

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**#include <hidef.h>**

**# include “derivative.h”**

**#include “scil.h”**

**Void main(void) {**

**Int n=0, ctr1=0, ctr2=0;**

**Char ARRAY[30];**

**SCI1\_Init(BAUD\_19200);**

**For(;;)**

**{ SCI1\_OutString (“Please Enter a sentence: ”);**

**SCI1\_InString(Array, 30);**

**SCI1\_OutString( Array+1);**

**While(Array[n]!\* 0x00) {**

**If(Array[n] == 0x20)**

**Ctr++;**

**Else**

**Ctr2++;**

**N++;**

**}**

**SCI1\_OutString (“ has”);**

**SCI1\_OutUDec(ctr2);**

**SCI1\_OutString (“characters &”) ;**

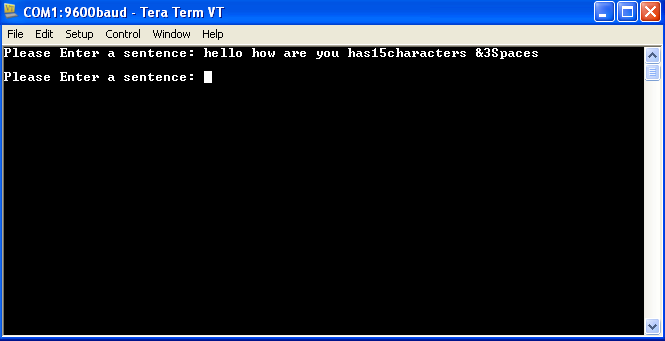
**SCI1\_OutString(ctr1);**

**SCI1\_OutString (“Spaces”);**

**SCI1\_OutChar (0x0A);**

**SCI1\_OutChar (0x0A);**

**SCI1\_OutChar (0x0D);**

****

# Conclusions and Recommendations

To conclude, the lab session was very useful in teaching the students how to use the Serial Terminal for input and output. In doing so, new skills were obtained as the programming was done in C. Using the Serial Monitor for input gives the programmer a variety options for the program given that the input has many forms; it can be a character, a string or even a float number. Taking such inputs from the board itself was an extremely difficult task. In addition to that, the Serial Terminal is capable of displaying outputs in a much neater and organized way. It can also represent many outputs, like characters for example, on the contrary to the board which is restricted to the space on the LCD screen and the LEDs.