



KHALIFA
UNIVERSITY

Khalifa University of Science, Technology and Research

Electronic Engineering Department

ELCE333: Microprocessor Systems Laboratory

Laboratory Experiment No. 5

Serial Communication Interface

Laboratory Partners

Anoud AL Shamsi - 100035514

Alya Humaid AlAlili -100037087

Afra Bin Fares - 100033139

Laboratory Instructor:

Mohammed Ali Saif Al Zaabi

Mahmoud Khonji

Spring 2015

Table of Contents

Abstract	3
1. Introduction	4
1.1 Aim	5
1.2 Objectives	5
2. Design and Results	6
Task 1: Transmit Characters	6
Task 2: Transmit String	6
Task 3: ASCII Data and Serial I/O	7
Task-4: Dip Switches Calculator	9
3. Conclusions and Recommendations	<i>Error! Bookmark not defined.</i>
4. Assignment Questions	<i>Error! Bookmark not defined.</i>

Abstract

The laboratory experiment five discusses the Serial Communication Interface in HCS12 microcontroller. Also, this laboratory experiment divides it into four main tasks. In the first task is about transmitting characters. Whereas, in the second task is about transmitting string. However, the third task is explains the ASCII Data and Serial I/O that's we receive the data from the keyboard and display it with its ASCII code on the serial terminal. Finally, the forth task aiming to implement a calculator that does the basic arithmetic operations that's the inputs will be taken from the Dip switches and the numbers and results are displayed on a serial terminal. Overall, the tasks were explained briefly in the report were the figures represents the outputs.

1. Introduction

A serial interface is a communication interface between two digital systems that transmits data as a series of voltage pulses down a wire where "1" is represented by a high logical voltage and a "0" is represented by a low logical voltage. The serial interface encodes the bits of a binary number by their "temporal" location on a wire rather than their "spatial" location within a set of wires. Encoding data bits by their "spatial" location is referred to as a parallel interface and encoding bits by their "temporal" location is referred to as a serial interface. the figure below shows the difference between the 2 interfaces.

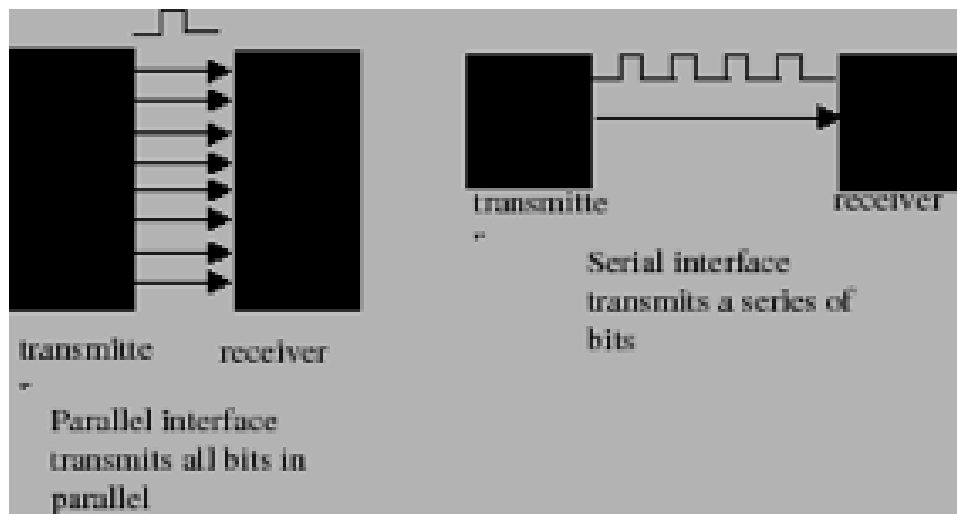


Figure 1: Difference between parallel and serial interfaces

The issue with a serial interface knows where the data is on the wire. The answer lies in creating a protocol. A protocol is an agreement between two parties about how the two parties should behave. A communication protocol is a protocol about how two parties should speak to each other. Serial communication protocols assume that bits are transmitted in series down a single channel. In an asynchronous serial interface (SCI) data is transmitted in frames. A frame is a complete packet of bits. The frame includes both information and control bits. In asynchronous serial protocols the frame often consists of a single start bit, data bits, parity bits, and sometimes a stop bit. A frame diagram is shown in the figure below. The start bit is used to signal the beginning of a frame and the stop bit signals the end of the frame. The parity bit is a special bit

that is used to detect transmission errors. The SCI interface is asynchronous because both devices do not need to synchronize their clocks before communicating. The receiver waits for the start bit and then begins reading the data line at the agreed upon baud rate.

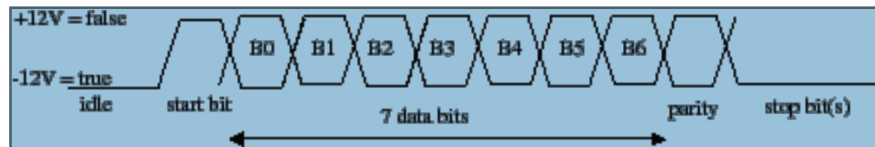


Figure 2: Frame representation

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, Results and Analysis

In this part of the report the tasks performed in the experiment will be discussed by explaining the steps of these tasks, listing, explaining and analyzing the results obtained from this experiment. The experiment is basically divided into four tasks. All four tasks were performed using c language written in the code warrior software, Tera Term display and the Dragon Plus Trainer Board.

2.1. TASK-1: Transmit Characters

In this part of the experiment the program below was run on code warrior display:

```
#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 */
}
```

Observations: the task was successfully done were it was observed on the Tera term display the ‘*’ character in every new line continuously.

2.2. TASK-2: Transmit String

In this part of the experiment, the program in task 1 was modified to transmit the following string

“I love Microcontrollers” continuously. Each string transmission was displayed on the terminal on the start of a new line. The program below was written in order for this task to be done:

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

Observations: the task was successfully done were it was observed on the Tera term display the string ‘I love Microcontrollers’ in every new line continuously.

Task-3 ASCII Data and Serial I/O

In this part of the experiment, a program was created that does the following:

1. Prints out the following text: The ASCII code for
2. Allows the user to type a single character on the keyboard.
3. Prints out the following text: is
4. Prints out the ASCII code for the character typed.
5. Prints a carriage return, line feed and repeats starting at step a.
6. If the user hits the Esc instead of the character the string “Bye” should be displayed in a new line and the transmission session should be halted.

The following code was used for this task:

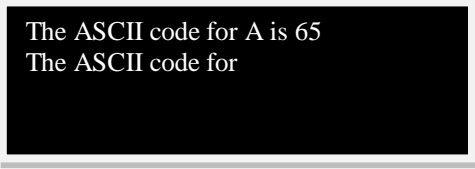
```

#include <hidef.h> /* common defines and macros */
#include "derivative.h" /* derivative-specific definitions */
#include "sci1.h"
void main(void) {
char a;
/* put your own code here */
SCI1_Init(BAUD_9600);
for(;;){
SCI1_OutString ("The ASCII code for "); // transmits * to SCI1
a=SCI1_InChar();//call function from sci1.h that takes in the characters
if(a==27){ // if the input equals to 27 which is the decimalequivalent to ESC key in the ASCII table
    SCI1_OutString ("bye"); // print out bye
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
    break;
} else{
    SCI1_OutChar(a);
    SCI1_OutString (" is ");
    SCI1_OutUDec(a);

    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
}
/* loop forever */
/* please make sure that you never leave main */
}
}

```

Observations: the task was successfully done . the program interacted as it was required to,with the user, where is looked like the following:



```

The ASCII code for A is 65
The ASCII code for

```

The program was tested in order to make sure that it is working for a variety of situations.

2.4. Task-4: Dip Switches Calculator

In this part of the experiment, it was required to write a C language program that reads two numbers from the 8-Dip switches. Where it does the following:

- 1- Each number is 4-bits.
- 2- The lower four DIP switches represent the No.1 and the higher four DIP switches represent the No.2.
- 3- The numbers should be displayed on the Serial Terminal.
- 4- Then the user should be prompted to enter an operation (+, -, /, *).
- 5- The entered operation should be performed on the two numbers
- 6- And finally the result should be displayed on Serial Terminal.

The following code was used for this in order to create this Dip switches calculator:

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

void main(void) {
    float n1,n2;
    float r;
    char a;
    SCI1_Init(BAUD_9600);
    DDRP=0xFF;          // Make PORTP output
    DDRH=0x00;          // assign it as an input - switches

    for(;;) {
        n1=PTH_PTH0*1+PTH_PTH1*2+PTH_PTH2*4+PTH_PTH3*8; // reading the 4 lower DIP switches and convert
        them to decimal
        n2=PTH_PTH4*1+PTH_PTH5*2+PTH_PTH6*4+PTH_PTH7*8; // reading the 4 higher DIP switches and convert
        them to decimal
        SCI1_OutString ("n1= "); // transmits n1= to SCI1
        SCI1_OutUDec(n1); // transmits the decimal value of n1
        SCI1_OutString (" n2= "); // transmits n2= to SCI1
        SCI1_OutUDec(n2); // transmits the decimal value of n2
        SCI1_OutChar (0x0D); // carriage return
        SCI1_OutChar (0x0A); // new line
        SCI1_OutString ("enter an operation (+,-,/,*)"); // transmits the string
        a=SCI1_InChar(); // read the char from input
        // the rest of the program in the next page
    }
```

```

if(a=='+'){//conditions here
    SCI1_OutChar(a);// transmits the inputted char to display we can either use + or the decimal equivalent in
    ASCII table
    r=n1+n2; // perform the desired operation and store it in r
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
    SCI1_OutUDec(n1); //transmits the decimal n1
    SCI1_OutString ("");//transmits the string +
    SCI1_OutUDec(n2); //transmits the decimal n2
    SCI1_OutString ("=");//transmits the string =
    SCI1_OutUDec(r); //transmits the decimal value of r
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
} else if(a=='-'){ // if -
    SCI1_OutChar(a); // transmits the inputted char to display
    r=n1-n2; // perform the desired operation and store it in r
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
    SCI1_OutUDec(n1); //transmits the decimal n1
    SCI1_OutString ("-");//transmits the string -
    SCI1_OutUDec(n2); //transmits the decimal n2
    SCI1_OutString ("=");//transmits the string =
    SCI1_OutUDec(r); //transmits the decimal value of r
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
} else if(a=='/'){ //if /
    SCI1_OutChar(a); // transmits the inputted char to display
    r=n1/n2; // perform the desired operation and store it in r
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
    SCI1_OutUDec(n1); //transmits the decimal n1
    SCI1_OutString ("/");//transmits the string /
    SCI1_OutUDec(n2); //transmits the decimal n2
    SCI1_OutString ("=");//transmits the string =
    SCI1_OutFloat(r,n2,n1); //transmits the float value of r
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
} else if(a=='*'){ //if *
    SCI1_OutChar(a); // transmits the inputted char to display
    r=n1*n2; // perform the desired operation and store it in r
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
    SCI1_OutUDec(n1); //transmits the decimal n1
    SCI1_OutString ("*");//transmits the string *
    SCI1_OutUDec(n2); //transmits the decimal n2
    SCI1_OutString ("=");//transmits the string =
    SCI1_OutUDec(r); //transmits the decimal value of r
    SCI1_OutChar (0x0D);// carriage return
    SCI1_OutChar (0x0A);// new line
}
}
}

```

Observations: the task was successfully done. The program interacted as it was required to, with the user, where is looked like the following:

No.1= 8 No.2= 15

Enter an operation (+, -, /, *) 8+15=23

the program was tested in order to make sure that it is working for a variety of situations.

3. Conclusion & Recommendations

In conclusion, the objectives of this experiment achieved successfully and now we are familiar with using an asynchronous serial communication interface in HCS12 microcontroller. Also, we understood the process of sending and receiving data from and to the PC using pre-written C library functions as well as serial I/O parameters. Moreover, we used a predefined functions in order to get us familiar with serial terminal. Although, we are able to designed a program that read the ASCII code and display it on the serial terminal and a DIP switches calculator that use both DIP switches and the PC keyboard. Overall, the experiment goals achieved successfully without facing any problems.

Assignment Questions

1)

```
#include <hidef.h>
#include "derivative.h"
#include "SCI1.h"

void main(void) {

    SCI1_Init(BAUD_19200);

    for(;;) {
        SCI1_OutChar ('*');
        SCI1_OutChar (0x0A);
        SCI1_OutChar (0x0D);
        return
    }
}
```

2)

```
#include <hidef.h>
#include "derivative.h"
#include "SCI1.h"

void main(void) {
    char sentence[100];
    char word;
    int count;
    int len;
    SCI1_Init(BAUD_9600);
    count=0
    for(;;) {
        SCI1_OutString ("Enter a sentence:");
        word= SCI1_InString(sentence, 500);
        SCI1_OutString (word);
        length=strlen(sentence);
        while(sentence[count] != 0x00)
            len++;
        if(sentence[count]!=0x20)
            count++;

        /*if (sentence[count]==0x20
            _count++; */
        SCI1_OutString (sentence);
        SCI1_OutString (" = ");
        SCI1_OutUDec (count);
        SCI1_OutString (" characters & ");
        SCI1_OutUDec (len-count);
        //SCI1_ OutUDec (_counter);
        SCI1_OutString (" spaces ");
        SCI1_OutChar (0x0A);
        SCI1_OutChar (0x0D);
    }
}
```