İ.T.Ü.
Faculty of Computer and Informatics
Computer Engineering

MICROCOMPUTER LAB REPORT

Lab No	**:** 07
Lab Date	**:** 14.11.2013
Group	**:** B09
Group Members	**:** 040100014 Teoman Turan 040100018 Mustafa Durmuş 040100117 Tuğrul Yatağan 040100124 Emre Gökrem

Research Assistant : Hasan Ünlü

1. THE AIM/CONTENT of THE EXPERIMENT

The purpose of doing this experiment is to take information about serial communication mechanism of CSM12C32, and to learn what UART, serial communication and baud rate are. For this experiment, PUTTY, a serial terminal software, has been used to observe the serial communication between our input and the output.

2. EQUIPMENT

To write and compile the code on CodeWarrior, then to use PUTTY, one of the laboratory's desktop computers has been used. CSM12CS32 Kit has been connected to that computer. After the compilation of the code and closed CodeWarrior, the kit has been reset, and then Putty has been run.

3. EXPERIMENTS

3.1. Section 3: Usage of Serial Terminal (PUTTY)

CSM12C32 has one UART port that is used for both programming and serial terminal. For this use, the instructions for PUTTY given in the lab sheet has been followed. Baud Rate has been set to 9600, and correct COM port, COM1, has been selected.

3.2. Section 4: Experiment

3.2.1. Transmit Only by Button

The following code in C programming language has been written and compiled on CodeWarrior:

```
#include <hidef.h> /* common defines and macros */
#include <mc9s12c32.h> /* derivative information */
#pragma LINK_INFO DERIVATIVE "mc9s12c32"
unsigned char text[] = "HELLO WORLD!"; // message to be sent

void sci_init(void)
{
	SCIBDL = 0x34; /*Configure baud rate 9600 bps*/
	SCIBDH = 0x00; /*an SCI clock modulo of 8MHz*/
```

```
    SCICR1 = 0x00; /*8 data bits, no parity*/
    SCICR2 = 0x0C; /*Enable Tx, Rx Bits*/
}

void send_char(unsigned char value)
{ // This blue code fragment is expected from us.
    while(!(SCISR1 & 0x40));  // While serial is not busy
        SCIDRL = value; // sent character to data register
}

void print_string(unsigned char *ptr)
{ // This blue code fragment is expected from us.
    int i;
    for(i=0; i<13; i++) // Counting all characters in the string
    {
        send_char(ptr[i]); // Characters are sent one by one
    }
}

void main(void)
{
    DDRE &= 0xFE; // PortE.0 is input for SW1
    sci_init(); // Setup registers

    for (;;)
    {
        if(!(PORTE & 0x01)) // When SW1 is pressed
            print_string(text);
    }
}
```

Then, the current window, True-Time Simulator & Real Time Debugger has been closed. The kit connected to the computer has been reset, then PUTTY has been run. On the serial terminal interface, “HELLO WORLD!” has been printed when SW1 is pressed.

3.2.2. Echo Mode

The following code in C programming language has been written and compiled on CodeWarrior:

```
#include <hidef.h> /* common defines and macros */
#include <mc9s12c32.h> /* derivative information */
#pragma LINK_INFO DERIVATIVE "mc9s12c32"
unsigned char text[20]; // variable for storing message
static int count=0; // number of characters in message
unsigned char temp; // temporary character

void sci_init(void)
{
     SCIBDL = 0x34; /*Configure baud rate 9600 bps*/
     SCIBDH = 0x00; /*an SCI clock modulo of 8MHz*/
     SCICR1 = 0x00; /*8 data bits, no parity*/
     SCICR2 = 0x0C; /*Enable Tx, Rx Bits*/
}

void send_char(unsigned char value)
{ // This blue code fragment is expected from us.
     while(!(SCISR1 & 0x40)); // While serial is not busy
          SCIDRL = value; // sent character to data register
}

void print_string(unsigned char * ptr)
{ // This blue code fragment is expected from us.
     int i;
     for(i=0;i<count;i++)
     {
          send_char(ptr[i]); // sent characters one by one
     }
     count = 0;
}
```

```
void main(void)
{
    sci_init(); // setup registers
    for (;;)
    { // This blue code fragment is expected from us.
        if(SCISR1 & 0x20) // If reception flag is set
    {
            if(SCIDRL == 0x0D) // When enter is pressed
                print_string(text); // The text is printed.
            else // The program keeps receiving characters.
                text[count++] = SCIDRL; // receive
                // character from data register
        }
    }
}
```

Then, the current window, True-Time Simulator & Real Time Debugger has been closed. The kit connected to the computer has been reset, then PUTTY has been run. On the serial terminal interface, we have written something, then pressed *enter*. As required for the Echo Mode, when *enter* has been pressed, the string we have written until *enter* being pressed has been exactly printed.