



MANHATTAN  
COLLEGE

# Lab 3: UART

ECHO!

School of Engineering

Electrical and Computer  
Engineering Department

ECEG 721-61 Embedded  
Systems

Jamie Quinn & Lucia  
Rosado-Fournier

November 6, 2020



## Table of Contents

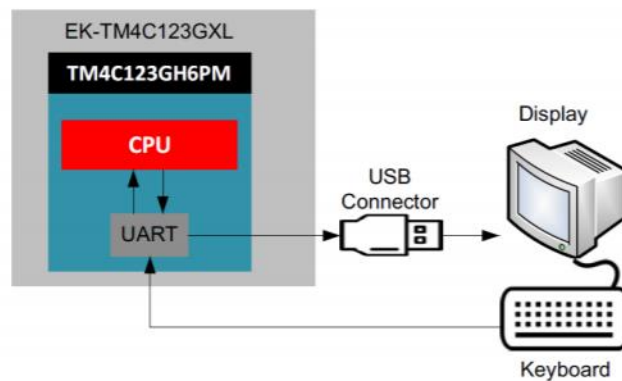
<b>1.1 Objective .....</b>	<b>3</b>
<b>1.2 Introduction.....</b>	<b>3</b>
<b>1.3 Component Requirements .....</b>	<b>3</b>
<b>1.4 Software .....</b>	<b>4</b>
Flowchart.....	5
C code.....	5
<b>1.5 Procedure.....</b>	<b>6</b>
<b>1.6 Observation .....</b>	<b>7</b>
<b>1.7 Summary.....</b>	<b>7</b>
<b>1.8 Exercise .....</b>	<b>7</b>
Ex #1 .....	7
Ex #2 .....	7

## 1.1 Objective

The main objective is to use the UART to send an echo of the data input back to the PC. We must also repeat the experiment, but change the baud rate, and find out what the maximum baud rate can be set in the UART peripheral of Tiva. This experiment gives us an understanding of the UART module and basics of serial communication to send and receive data.

## 1.2 Introduction

In this experiment, we set up a Serial Communication Interface module that transmits received data from an external device along with some message to indicate the reception. The UART is connected to the serial port of the PC that runs over the debug USB cable as shown in Figure 1. Through serial communication, data is received by the UART from the USB cable and displayed on the serial monitor (Tera Term).



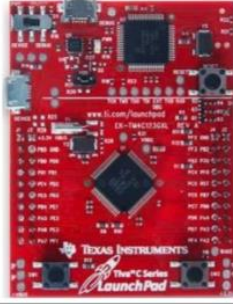
*Figure 1 – Functional Block Diagram Experiment.*

## 1.3 Component Requirements

### 1.3.1 Software Requirements

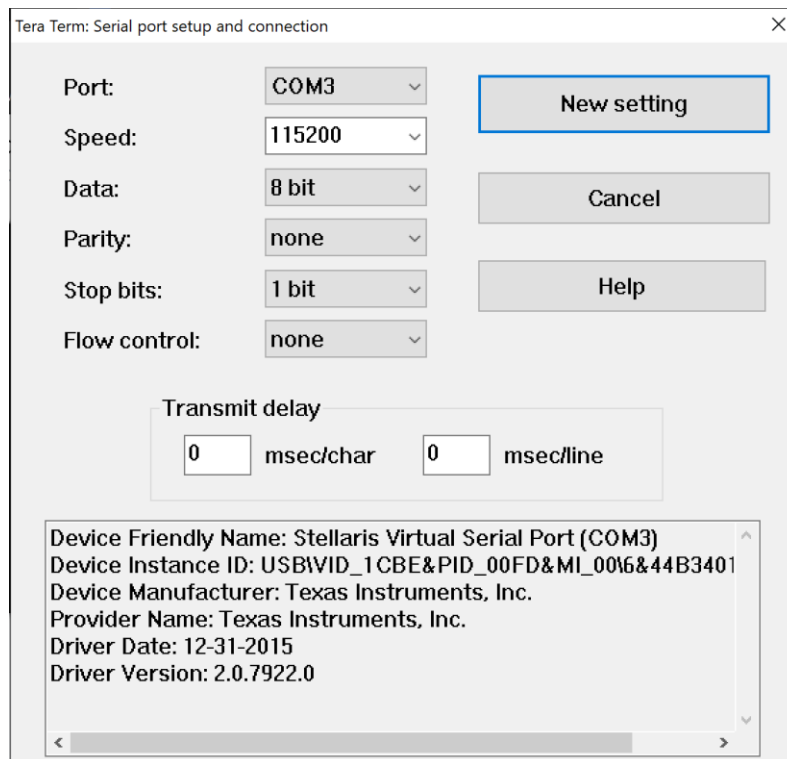
1. Keil
2. Tera Term
3. TivaWare\_C\_Series

### 1.3.2 Hardware Requirements

S.No	Components	Specifications	Images
1.	Tiva LaunchPad	EK-TM4C123GXL LaunchPad	
2.	USB cable		

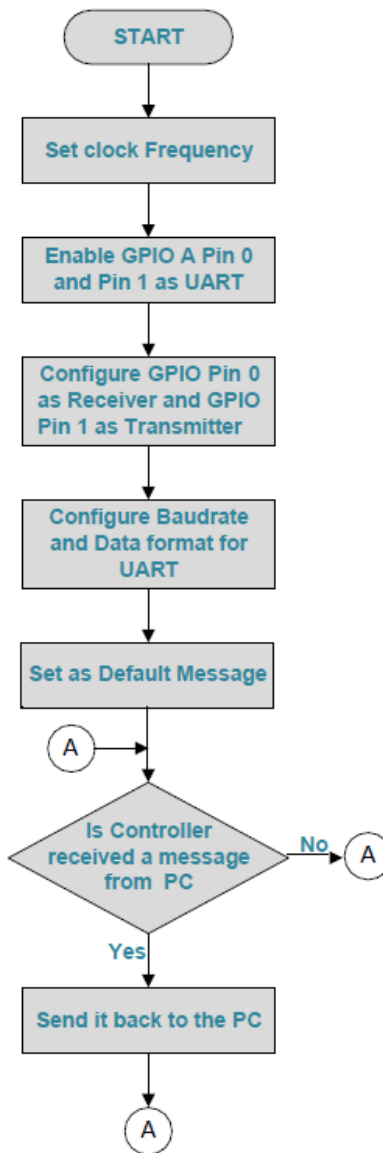
### 1.4 Software

The software for the experiment is written in C and developed using the Keil Integrated Development Environment (IDE). The software is programmed into the target device TM4C123GH6PM on the EK-TM4C123GXL using the USB interface. The software Tera Term is also used to communicate with the serial port on the board. The setup necessary can be seen in the image below and will be elaborated on in the procedure section.



*Figure 2 – Serial Port Setup on Tera Term*

### Flowchart



**Figure 3** – Flowchart to use UART.

### C code

```

#include <stdint.h>
#include <stdbool.h>
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/gpio.h"
#include "driverlib/pin_map.h"
#include "driverlib/sysctl.h"
#include "driverlib/uart.h"
//Holds the receive and transmit hex values to be used later to configure
the GPIO Pin

```



```
#define GPIO_PA0_U0RX 0x00000001
#define GPIO_PA1_U0TX 0x00000401
int main(void) {
    //Set clock rate
    SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|
SYSCTL_XTAL_16MHZ);
    //Enable and configure peripherals
    SysCtlPeripheralEnable(SYSCTL_PERIPH_UART0);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);
    GPIOPinConfigure(GPIO_PA0_U0RX);
    GPIOPinConfigure(GPIO_PA1_U0TX);
    //Configure type
    GPIOPinTypeUART(GPIO_PORTA_BASE, GPIO_PIN_0 | GPIO_PIN_1);
    //Configure the UART below
    UARTConfigSetExpClk(UART0_BASE, SysCtlClockGet(), 115200,
(UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE | UART_CONFIG_PAR_NONE));
    //Echo UART characters
    UARTCharPut(UART0_BASE, 'E');
    UARTCharPut(UART0_BASE, 'c');
    UARTCharPut(UART0_BASE, 'h');
    UARTCharPut(UART0_BASE, 'o');
    UARTCharPut(UART0_BASE, ' ');
    UARTCharPut(UART0_BASE, '0');
    UARTCharPut(UART0_BASE, 'u');
    UARTCharPut(UART0_BASE, 't');
    UARTCharPut(UART0_BASE, 'p');
    UARTCharPut(UART0_BASE, 'u');
    UARTCharPut(UART0_BASE, 't');
    UARTCharPut(UART0_BASE, ': ');
    UARTCharPut(UART0_BASE, ' ');
    UARTCharPut(UART0_BASE, '\n');
    //Consistently read in values the user puts in and echo them back
out
    while(1){
        if (UARTCharsAvail(UART0_BASE)) UARTCharPut(UART0_BASE,
UARTCharGet(UART0_BASE));
    }
}
```

### 1.5 Procedure

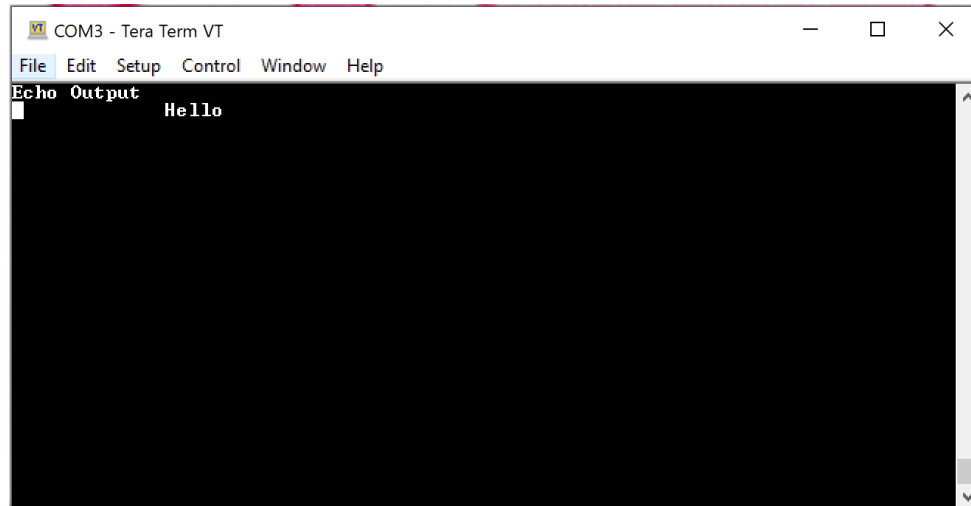
The procedure to complete this lab is as follows:

1. Connect the microcontroller to the computer using the USB cable. Make sure it is connected to the debug port on the board and the switch is flipped to debug as well.
2. Add the necessary files and libraries to the project to ensure the libraries and files referenced in the main.c can run, like it was done in the previous lab.
3. Build and load the code to the board, ensuring there are no errors. Then press the reset button on the board.

4. Open Tera Term UART terminal window and configure the COM port and baud rate to match the parameters on your personal machine. Following the code, the baud rate was 115200.
5. Type characters on the keyboard and observe the terminal window.

### 1.6 Observation

As expected, the terminal shows “Echo Output: “ when the code runs for the first time. Afterwards, as characters were typed on the keyboard, said characters were echoed onto the terminal. The output of this can be seen in the image below.



*Figure 3 - Tera Term Output.*

Overall, there were no issues with running the program and using Tera Term with it.

### 1.7 Summary

Overall, the experiment was simple to implement, just needing a few extra steps as compared to past experiments. It took about two hours altogether to complete the experiment.

### 1.8 Exercise

*Ex# 1 Change the baud rate to 19200 and repeat the experiment.*

To change the baud rate, everything in the code stays the same except for the following line.

```
UARTConfigSetExpClk(UART0_BASE, SysCtlClockGet(), 19200,
(UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE | UART_CONFIG_PAR_NONE));
```

The baud rate also needs to be changed on Tera Term. If the baud rates don't match up, the output will be unintelligible text.

*Ex# 2 What is the maximum baud rate that can be set in the UART peripheral of Tiva?*

As stated by the Tiva TM4C123GH6PM Microcontroller data sheet, the maximum baud rate is 10Mbps.