Lab 10: Bluetooth Communication

Name: Zaid Bin Khalid ID: zk08128 Section: T1

Objective

To build wireless communication channel between TivaC TM4C123 and Android device using Bluetooth module HC-05

In-Lab

Task 1: Establish Bluetooth communication between TivaC and Android to transmit and receive characters

Task 2: To display different colors on RGB LED and control servo using Bluetooth

Task 3: LED Control with HC-05 and Andriod App using Keil uvision

1 Bluetooth Module HC-05

Wireless communication is swiftly replacing the wired connection when it comes to electronics and communication. Designed to replace cable connections HC-05 uses serial communication to communicate with the electronics. Usually, it is used to connect small devices like mobile phones using a short-range wireless connection to exchange files. It uses the 2.45GHz frequency band. The transfer rate of the data can vary up to 1Mbps and is in range of 10 meters. The HC-05 module can be operated within 4-6V of power supply. It supports baud rate of 9600, 19200, 38400, 57600, etc.

This module can be used to perform low-cost wireless communication between microcontrollers such as Arduino, TM4C123, Pic Microcontroller, Raspberry Pi, BeagleBone, STM32, and Arduino Mega, etc. Also, we can use it for communication between a microcontroller and PC or Android application. There are many Android applications available in the Google Play store which we can use to send and receive data to/from HC05. The pin configuration for this module is given in Fig. 7 with description given below.

- Enable (Key) - This pin is used to set the Data Mode or and AT command mode (set high)
- VCC - This is connected to +5V or +3.3V power supply

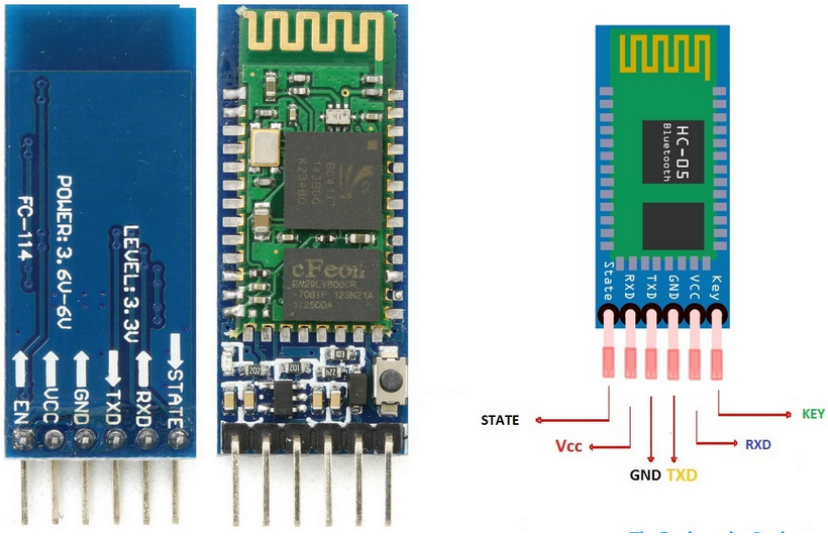


Figure 1: Pin Configuration for HC-05

- Ground - Connected to ground of powering system
- Tx (Transmitter) - This pin transmits the received data Serially
- Rx (Receiver) - Used for broadcasting data serially over Bluetooth
- State - Used to check if the Bluetooth is working properly

2 UART communication in TM4C123GH6PM

UART stands for Universal Asynchronous Receiver/Transmitter. It's not a communication protocol like SPI and I2C, but a physical circuit in a microcontroller, or a stand-alone IC. A UART's main purpose is to transmit and receive serial data using just two wire TX and RX.

UART Module	Rx Pin	Tx Pin
UART0	PA0	PA1
UART1	PC4	PC5
UART2	PD6	PD7
UART3	PC6	PC7
UART4	PC4	PC5
UART5	PE4	PE5
UART6	PD4	PD5
UART7	PE0	PE1

Figure 2: UART Channels in TM4C123GH6PM

HC-05 is a two-way full-duplex Bluetooth to serial module. This Bluetooth device operates on UART communication. In other words, it uses serial communication to transmit and receive data serially over standard Bluetooth radio frequency.

UART communication port of the TM4C123GH6PM microcontroller is used with HC-05. This microcontroller has on-chip 5 UART ports such as UART0 to UART5. We can use any one of these UART ports to interface TM4C123 with the BT device. The table in Fig. 2 shows all UART ports of TM4C123 Tiva launchpad and their corresponding GPIO pins.

3 Android Bluetooth Terminal App

In order to establish Bluetooth communication between TivaC and Android device, we will use a android application named ”Serial Bluetooth Terminal”, freely available on Play Store app. You are required to download and install that application. Once the application would be downloaded, and our HC-05 connections are made with TivaC in Task 01 below, we will follow the steps shown in Fig. 3 to send and receive commands over Serial monitor of the app and the Energia.

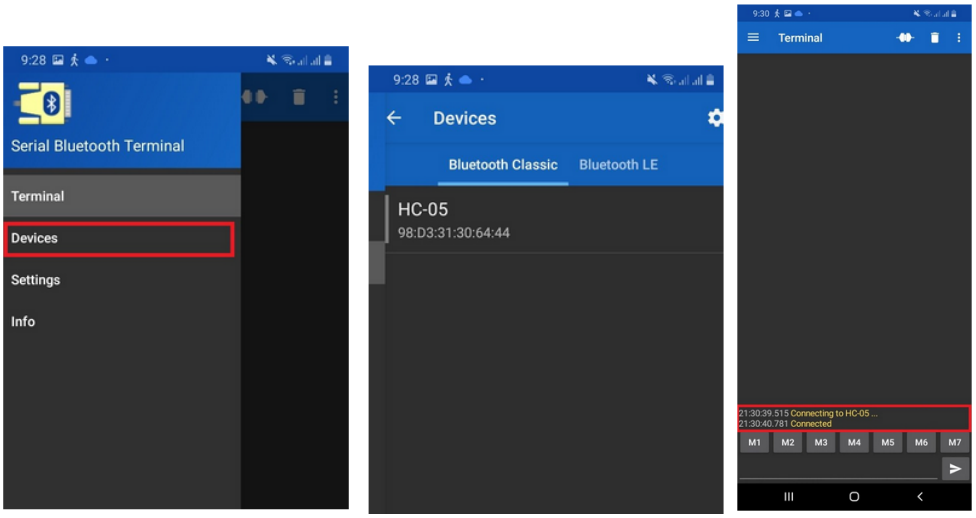


Figure 3: Steps for Serial Bluetooth Terminal App

4 TivaC LaunchPad with Energia

In Energia IDE, unlike Kiel uVision, we can use TivaC LaunchPad pins for various peripherals without the need to activate Ports using registers or specifying function of the pin. But it also comes with limitations of usability and programmable scope of the board. In Energia, we can refer to Pins of TivaC directly using numeric digit like 1,2,3.. and so on. Pin map for the EK-TM4C123GXL LaunchPad is given in Fig. 4 with Black Columns under J1, J2, J3 and J4.

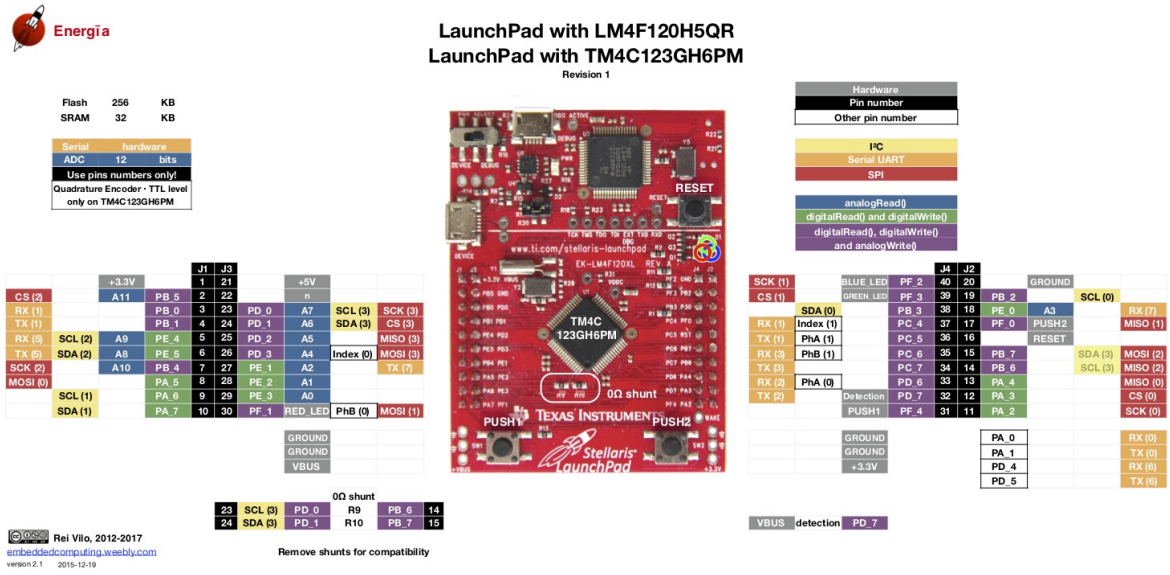


Figure 4: TivaC Pin Distribution with Energia

In-Lab Tasks

Task 1: Establish Bluetooth communication between TivaC and Android to transmit and receive characters

In this task, we are going to establish simple Bluetooth communication using UART0 of TivaC and send characters and string from Android device to be displayed on Energia Serial Monitor. Follow the step below to achieve the communication:

- Make the connections given in Fig. 5.
- Download and open ”bluetooth.ino” in Energia.
- Modify the code where you are required to in order to make it functional.
- Download and Install ”Serial Bluetooth Terminal” in your Android Mobile Device.
- Connect your mobile device with Bluetooth module using steps given in Fig. 3 with password ”1234” or ”0000”. The Bluetooth device name is mentioned on module itself.
- Once a successful communication is established, you should be able to send and receive the characters using Serial.

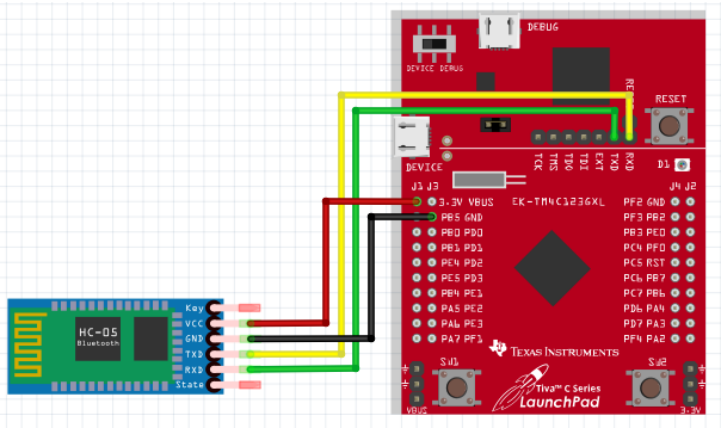


Figure 5: Circuit Connection between TivaC and HC-05

Write your brief and complete understanding of how the code works. Explain how Bluetooth module use UART protocol to communicate?

The code initializes UART communication on TivaC's UART0 and connects it to the HC-05 Bluetooth module. The HC-05 module uses UART to send and receive data serially. Commands sent from an Android device using a Bluetooth terminal app are received by TivaC and displayed on the Energia Serial Monitor.

Provide your clear circuit image below.

Provide your code here with appropriate comments below (Get the circuit demonstration checked with RA within Lab)

```
#include <Servo.h>
Servo myservo;  //Creating a Servo Object
int pos = 0;    //Position of Servo
const int servoPin = 26; //Pin at PD3

//Define the RGB pins
const int red = 30;     //Present at PF1
const int green = 40;   //Present at PF2
const int blue = 39;    //Present at PF3

int colorDetect=0;     //For different color detection
char received;         //Characters to be send/receieve on UART

void setup(){
  //Initialize Baud Rate at 9600 below
  Serial.begin(9600);
  //Specify pin modes for red, green and blue pins below
  pinMode(red, OUTPUT);   //Red color pwm pin defined as output
  pinMode(green, OUTPUT); //Green color pwm pin defined as output
  pinMode(blue, OUTPUT);  //Blue color pwm pin defined as output

  //We use analogWrite() to control PWM at each pin PF1, PF2 and PF3
  //There differnt combination would generate different colors for value between 0-255
  //Starting with no color on RGB LEDs
  analogWrite(red,0);
  analogWrite(green,0);
  analogWrite(blue,0);

  //Servo to be connected at PD3
  myservo.attach(servoPin);
 }


void loop(){
  //Following Condition is true whenever we send a command from Bluetooth Terminal App
  if(Serial.available()>0){

    //Read the bluetoot data and store it in colorDetect variable using Serial.read() command here
    colorDetect = Serial.read();

    //Following lines help debugging by printing whatever character recieved on Serial Monitor
    char Rec = char(colorDetect);
    if (Rec != '0')
    {
    Serial.println(Rec); //This is to visualise the received character on the serial monitor
    }
  }

  //Following function should take character received on bluetooth as argument and should correspondingly change
servo position and RGB color
  chooseColor(colorDetect);
}
```

Task 2: To display different colors on RGB LED and control servo using Bluetooth

In this task, we will control the color of the built-in RGB LED of TivaC and position of servo according to command received by Android Terminal. In order to achieve the above, follow the instructions below:

- Connect Servo's VCC, GND and PWM pins to 3.3V/Bus, GND and PD3 of the TivaC Board.
- Initialize pinModes for Servo PWM, and built-in RGB LED which is present on PortF at pins PF1, PF2 and PF3.
- Use myservo.write(x) to rotate servo to position x.
- Use analogWrite() to obtain different colors on built-in RGB LED.

Using the above steps, you are required to achieve the following in this task:

- Servo to be moved 45 degrees with color RGB color RED on command "A"
- Servo to be moved 90 degrees with color RGB color GREEN on command "B"
- Servo to be moved 135 degrees with color RGB color BLUE on command "C"
- Servo to be moved 180 degrees with color RGB color WHITE on command "D"

Write your brief and complete understanding of how the code works for controlling servo and RGB color using Bluetooth communication.

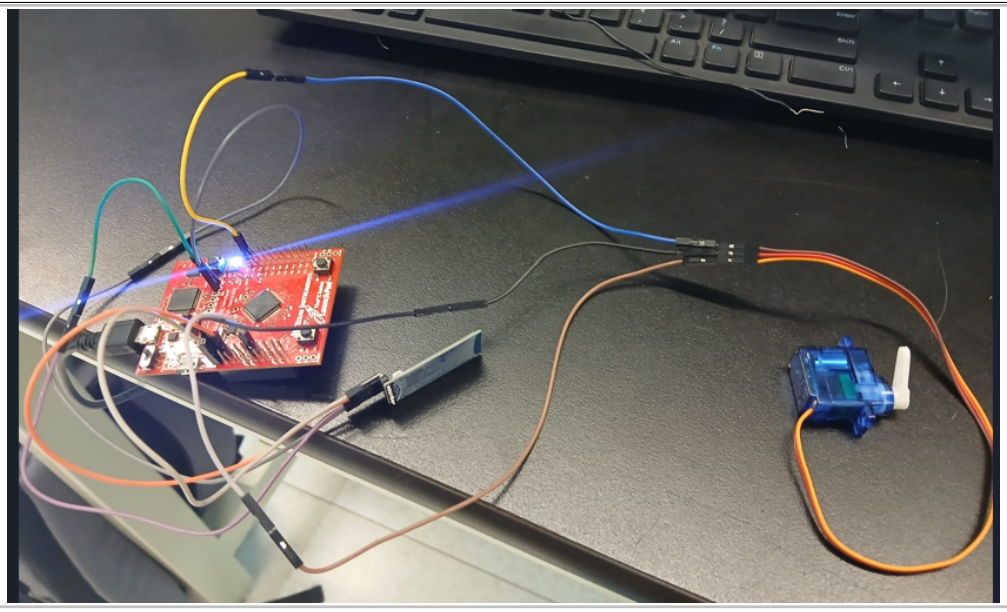
The code configures the RGB LED on the TivaC LaunchPad and a servo motor. Bluetooth commands from an Android device control the color of the LED and the servo's position. Each command adjusts the RGB LED color and moves the servo to a specific angle, demonstrating how Bluetooth communication can control multiple peripherals.

Provide your code here with appropriate comments below (Get the circuit demonstration checked with RA within Lab)

```
#include <Servo.h>
Servo myservo;  //Creating a Servo Object
int pos = 0;    //Position of Servo
const int servoPin = 26; //Pin at PD3

//Define the RGB pins
const int red = 30;     //Present at PF1
const int green = 40;   //Present at PF2
const int blue = 39;    //Present at PF3

int colorDetect=0;     //For different color detection
char received;         //Characters to be send/receieve on UART

void setup(){
  //Initialize Baud Rate at 9600 below
  Serial.begin(9600);
  //Specify pin modes for red, green and blue pins below
  pinMode(red, OUTPUT);   //Red color pwm pin defined as output
  pinMode(green, OUTPUT); //Green color pwm pin defined as output
  pinMode(blue, OUTPUT);  //Blue color pwm pin defined as output

  //We use analogWrite() to control PWM at each pin PF1, PF2 and PF3
  //There differnt combination would generate different colors for value between 0-255
  //Starting with no color on RGB LEDs
  analogWrite(red,0);
  analogWrite(green,0);
  analogWrite(blue,0);

  //Servo to be connected at PD3
  myservo.attach(servoPin);
 }

void loop(){
  //Following Condition is true whenever we send a command from Bluetooth Terminal App
  if(Serial.available()>0){

    //Read the bluetoot data and store it in colorDetect variable using Serial.read() command here
    coloDetect = Serial.read();

    //Following lines help debugging by printing whatever character recieved on Serial Monitor
    char Rec = char(colorDetect);
    if (Rec != '0')
    {
    Serial.println(Rec); //This is to visualise the received character on the serial monitor
    }
  }

  //Following function should take character received on bluetooth as argument and should correspondingly change servo position and RGB color
  chooseColor(colorDetect);
}

void chooseColor(char color){
  //Check condition for character stored in "color" and perform your task 02 here
  if (color = 'w')
  {
    analogWrite(red,255);
    analogWrite(green,255);
    analogWrite(blue,255);
    myservo.write(0);
  }
  else if (color = 'r')
  {
    analogWrite(red,255);
    analogWrite(green,0);
    analogWrite(blue,0);
    myservo.write(30);
  }
  else if (color = 'g')
  {
    analogWrite(red,0);
    analogWrite(green,255);
    analogWrite(blue,0);
    myservo.write(45);
  }
  else if (color = 'b')
  {
    analogWrite(red,0);
    analogWrite(green,0);
    analogWrite(blue,255);
    myservo.write(60);
  }
  else if (color = 'o')
  {
    analogWrite(red,102);
    analogWrite(green,0);
    analogWrite(blue,153);
    myservo.write(90);
  }
}
```

Attach Screen-shot of Energia Serial Monitor Below

Provide your clear circuit image below.



Task 3: LED Control with HC-05 and Andriod App using Keil uvision

TM4C123 Tiva Launchpad has an RGB LED connected with PF1, PF2, and PF3 pins. We will send ON/OFF commands from an Android application. Based on these commands, LEDs will turn on or off.

Commands used to control onboard LEDs of Tiva Launchpad are listed below:

Command From App	Function
A	Green LED ON
B	Green LED OFF
C	RED LED ON
D	RED LED OFF
E	BLUE LED ON
F	BLUE LED OFF

Figure 6: Commands for LEDs

For example, if we send the character 'A' from the Android app to HC-05 the green LED of Tiva Launchpad will turn on. Similarly, if we transmit 'B' from the Android app to HC-05 green LED of Tiva Launchpad will turn off. Likewise, commands for the other two LEDs are mentioned in the above table.

- We will use a UART5 port. For the UART5 port, the PE4 GPIO pin is a receiver pin (Rx) and the PE5 pin is a transmitter pin (Tx). Now make connections with HC-05 Bluetooth module and Tiva Launchpad according to schematic diagram (Figure 7).
- Now create a new project in Keil uvision and compile the skeleton code (will be given in modules section on LMS). After that upload the generated hex file to Tiva Launchpad.
- Modify the code where you are required to in order to make it functional.
- Download and Install "Serial Bluetooth Terminal" in your Android Mobile Device.
- Connect your mobile device with Bluetooth module using steps given in Fig. 3 with password "1234" or "0000". The Bluetooth device name is mentioned on module itself.
- Once a successful communication is established, you should be able to send and receive the characters using Serial.

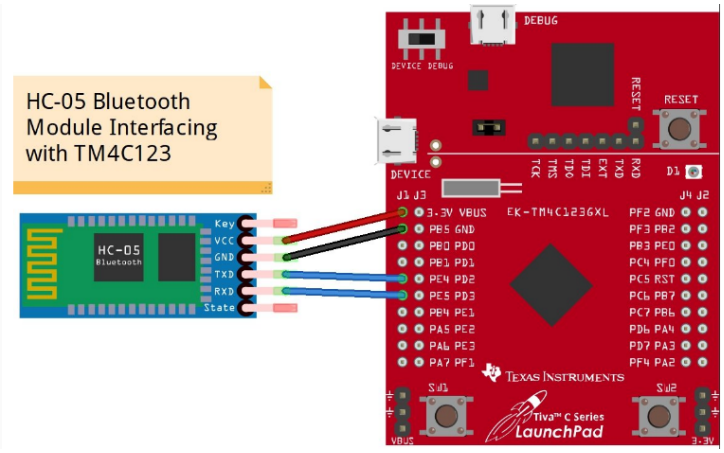


Figure 7: Circuit Connection

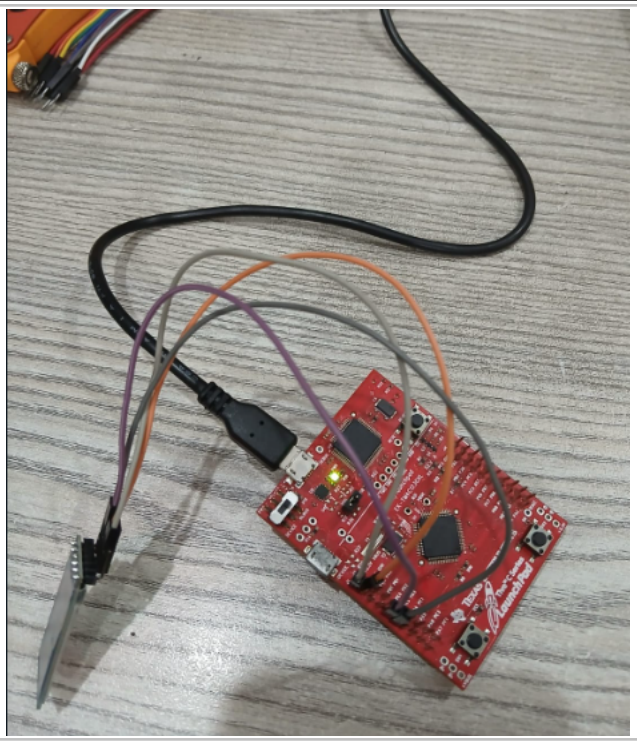
Write your brief and complete understanding of how the code works.

The code in Keil uVision uses UART5 on the TivaC LaunchPad to communicate with the HC-05 Bluetooth module. Commands from an Android app control the onboard RGB LEDs by toggling the pins connected to each LED color. Each received command turns an LED on or off, providing basic Bluetooth-based remote control.

Provide your code here with appropriate comments below (Get the circuit demonstration checked with RA within Lab)

```
#include "TM4C123.h"
#include <stdint.h>
#include <stdlib.h>
void Delay(unsigned long counter); // used to add delay
void HC05_init(void); // Initialize UART5 module for HC-05
char Bluetooth_Read(void); //Read data from Rx5 pin of TM4C123
void Bluetooth_Write(unsigned char data); // Transmit a character to HC-05 over Tx5 pin
void Bluetooth_Write_String(char *str); // Transmit a string to HC-05 over Tx5 pin
int main(void)
{

 HC05_init(); // call HC05_init() to initialze UART5 of TM4C123GH6PM

 /* Set PF1, PF2 and PF3 as digital output pins */

 SYSCTL->RCGCGPIO |= 0x20;   /* enable clock to GPIOF */
    GPIOF->DIR |= 0x7;          //set PF1, PF2 and PF3 as digital output pin
    GPIOF->DEN |= 0x7;          // CON PF1, PF2 and PF3 as digital GPIO pins
    Delay(200);

 while(1)
 {
  char c = Bluetooth_Read();          /* get a character from UART5 */

/* Check the received character and take action to control onboard LEDs of TM4C123 */
/* Send status string to Andriod app after turning on/off LEDs */

   if( c== 'a'){ // refer to the lab manual
   GPIOF->DATA |=(1<<2); //write down x
   Bluetooth_Write_String("---\n");
  }
  else if( c== 'b'){
   GPIOF->DATA &=(1<<2); //write down x
   Bluetooth_Write_String("---\n");
  }
  else if( c== 'c'){
   GPIOF->DATA |=(1<<1); //write down x
   Bluetooth_Write_String("---\n");
  }
  else if( c== 'd'){
   GPIOF->DATA &= (1<<1); //write down x
   Bluetooth_Write_String("---\n");
  }
  if( c== 'e'){
   GPIOF->DATA |=(1<<3); //write down x
   Bluetooth_Write_String("---\n");
  }
  else if( c== 'f'){
   GPIOF->DATA &=(1<<3); //write down x
   Bluetooth_Write_String("---\n");
  }
 }
}

void HC05_init(void)
{
 SYSCTL->RCGCUART |= 0x20;  /* enable clock to UART5 */
    SYSCTL->RCGCGPIO |= 0x10;  /* enable clock to PORTE for PE4/Rx and RE5/Tx */
    Delay(1);
    /* UART0 initialization */
```

Provide your clear circuit image below.



5 Assessment Rubrics

Marks distribution

		LR1	LR2	LR4	LR5	LR9
	Task 1	-	10 points	05 points	10 points	15 points
	Task 2	10 points	10 points	05 points	10 points	
In-lab	Task 3	-	10 points	05 points	10 points	
Total Marks 100						

Marks obtained

		LR1	LR2	LR4	LR5	LR9
	Task 1	-				
	Task 2					
In-lab	Task 3	-				
Total Marks 100						