Lab #9 Synchronized LEDs (UART)
ECE 266, Spring 2019

Lab location: 3264 SEL

Lab Time: Flexible

Demo Due: Your lab time in Week 15.

Introduction

This is a bonus lab. You may use it to make up any deficit in your lab grade. However, your overall lab grade will not be over 100% (i.e. it does NOT overflow). The total number of points for this lab is 60.

The goal of this lab is for you to utilize a form of serial communication between two microcontrollers, in this lab the UART hardware function of Tiva C, to build a distributed embedded system.

The learning objectives are:

- To learn how to program the UART module of Tiva C.
- To learn how to make two LaunchPads communicate through UART
- To explore the UART functions of the TivaWare.

Notes:

- You will study the details of UART (universal asynchronous transmitter/receiver), and two other forms of serial communications (I2C and CAN) in week 15. If you choose to do this lab, it helps you get the hand-on experience before those lectures.

Prelab

- Study "uart.c" in the Util/ folder (that contains the Util.lib). It implements the driver functions for the LaunchPad to communicate with PC through UART.
- Read the following pages of Ch. 31 UART, "TivaWare Peripheral Driver Library User Guide":
    - Page 559, Sec. 31.1, Introduction
    - Page 568, Sec. 31.2.15, UARTConfigSetExpClk
    - Page 565, Sec. 31.2.2.9, UARTCharPut
    - Page 564, Sec. 31.2.2.7, UARTCharGet
- Read pages 893-897 (to the end of Sec. 14.3.3) of Tiva C Data Sheet.

- Print out a copy of the lab grading form ("Lab9GradeForm.pdf"). Fill the form and give it to your TA before your demo.

Part 0: Preparation

In this lab, you need to use two LaunchPads of your group. Use one LaunchPad as a **local node**, and the other as a **remote node**. Attach each LaunchPad to its own Grove base. You need to make a crossover cable (see below) to connect the two LaunchPads, so that one LaunchPad's UART Tx (transmit) pin will connect to the other LaunchPad's UART Rx (receive) pin.

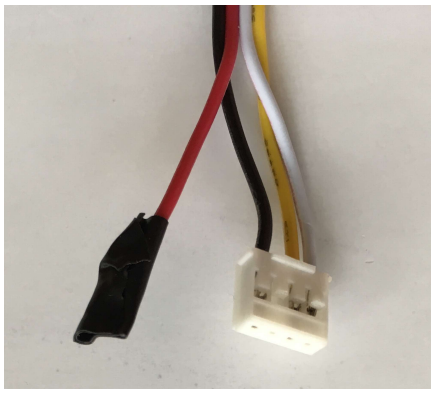
In addition, you need the following items before you start:

1. **Some eye protection**, e.g. protective eyewear.
2. A flathead precision screwdriver of 1.0mm or similar size. Example: https://www.amazon.com/Stanley-66-052-6-Piece-Precision-Screwdriver/dp/B00009OYGV/ref=sr_1_2?ie=UTF8&qid=1477753743&sr=8-2&keywords=precision+screwdriver+1.4mm
   If you don't have one, a pen with a small tip may work but you may spend some extra time.
3. Electrical tape.

Do the following:

1. Wear your eye protection.
2. Pick up a Grove cable in your Grove Start Kit.
3. Choose either terminal (in white color) of the cable. Look at the terminal, you will see four small tabs on one side of the terminal. Separate the four wires near the terminal by hand for about 1-1.5 inches.
4. Hold the terminal such that the four tabs face up. Locate the tab connecting to the **yellow** wire. Use the precision screwdriver, carefully pry and pop this tab *slightly* (do NOT break the tab), and pull out the yellow wire from the terminal.
5. Repeat step 2 but this time on the **white** wire, to pull it out from the terminal.
6. Repeat step 2 and this time on the **red** wire, to pull it out from the terminal.
7. Insert the **yellow** wire into the tab where the **white** wire was inserted.
8. Insert the **white** wire into the tab where the **yellow** wire was inserted.
9. Use electrical tap to wrap up the end of the **red** wire. Do NOT insert it back to any tab.
10. The black wire should stay intact.

Now you have a crossover cable. The following picture shows how the changed terminal should look like:



Use the crossover table to connect jumper **J11** of the two LaunchPads. This jumper connects to the UART1 unit of Tiva C. In this configuration, the Rx and Tx pins of UART1 are connected to PB0 and PB1 (Port B Pin 0 and Pin 1), respectively.

In your working directory for ECE 367, create a new folder named "Lab9". In this folder, create two sub-folders named "Lab9Local" and "Lab9Remote". You will create two CCS projects, for the local node and the remote node, respectively, in those two sub-folders. *As in previous labs, one partner should do this step and push the files into your GitHub repository, and the other partner(s) should pull the files from the GitHub repository. All in the group should discuss and work out a collaboration plan.*

In this lab, there is no template code provided. Instead, you may reuse the program files from previous labs and "uart.c" under Util/ of your working directory.

Part 1: Talk through UART

Make a copy of "uart.c" from Util/ to the Lab9Local/ folder and rename it to "uart1.c" Revise the functions to fit the need of this lab, and create a header file "uart1.h" for it. Rename all functions, e.g. uartInit() to uart1Init(), so as to avoid name conflicts between your UART functions and the UART functions in the Util.lib. You may delete any UART functions in "uart.c" that you do not need.

Copy "uart1.c" and "uart1.h" to Lab9Remote/ folder. If you further revise those two files, make the change synchronized in Lab9Local/ and Lab9Remote/.

Revise main.c in Lab9Local/ and Lab9Remote/, so that if the user pushes SW1 on the remote node, the local node will print a message "SW1 is pushed" on Putty; and if the user pushes SW2 on the remote node, the local node will print "SW2 is pushed".

Note: You have to download different programs to the local LaunchPad and the remote LaunchPad. To avoid confusing CCS, connect only one LaunchPad at a time to CCS. For the other LaunchPad, connect the USB cable to its USB port on the side (not the one on the top). That USB port is intended to provide power only to LaunchPad.

Alternatively, you and your partners may use connect the two LaunchPads to two different computers.

Hint: There are several high-level approaches to implementing a networked embedded system like this. To simplify, you may make the local node as a command node, and the remote node as a response node, such that the local node sends commands to the remote node (e.g., a one-byte code for checking SW1) and the remote node sends back a response (e.g., a one-byte code indicating if SW is pushed or not).

Part 2: Synchronized LED

Revise your program files for the local node and the remote node, so that a user can use the pushbuttons of the local LaunchPad to control the LEDs of both LaunchPads.

- Initially, the LEDs of both the LaunchPads should be turned off.
- If the user pushes SW1 of the remote LaunchPad, both LEDs should be turned on.
- If the user pushes SW2 of the remote LaunchPad, both LEDs should be turned off.

Demo your system to your TA. If you can demo Part 2, you don't have to demo Part 1. However, you are NOT suggested to skip Part 1.

Lab Submission

No lab report is required for this lab.

Submit 1) a zip file named **"Lab9.zip"** for all the source code files in your Lab9/ folder (including Lab9Local/ and Lab9Remote/ sub-folders). Do NOT include non-source code files in Lab9.zip.

Make sure that you have pushed the latest version of your code into GitHub. **This step is required for you to get the grade.**