DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
UNIVERSITY OF BRITISH COLUMBIA
CPEN 391 – Computer Systems Design Studio
Fall 2016/2017 Term 2

Exercise 1.6
Adding the Bluetooth Controller

Introduction

Bluetooth, like GPS, is also becoming ubiquitous in mobile computing. Bluetooth was originally invented as a "wireless alternative" to traditional serial connections that use physical wires, e.g. RS232. These days virtually every Laptop, tablet and mobile device has a Bluetooth chipset built in to communicate wirelessly with other devices. Many peripherals like headsets, keyboards, mice, game controllers, medical equipment etc. also have Bluetooth connectivity.

There's a simple intro/overview of Bluetooth here
https://learn.sparkfun.com/tutorials/bluetooth-basics

We'll talk about Bluetooth more when we get to the second half of this course, i.e. Module 2, where we will connect our NIOS system to an Android phone/tablet and use Bluetooth to communicate between them wirelessly. For this exercise, we are just going to connect a Bluetooth "dongle" to one of the serial ports we created in exercise 1.3.

The device we will be using comes from Sparkfun and uses their RN42 Bluetooth Mate "Silver" device https://www.sparkfun.com/products/12576 which has range of about 10m and costs about $25. They also do a "Gold" device with longer range up to 100m which costs $35.

You can clearly see in the image above that the device has a serial port connection via the RX_I and TX_O signals. You only need to connect these to a serial port and supply ground and a 5v supply to give you a wireless serial port (*note the CTS/RTS "flow control" signals can be left unconnected since by default they are disabled in the device*). The device can communicate at speeds between 2400 and 115kBaud.

The default configuration for this Bluetooth module is:

- Bluetooth "slave" mode i.e. it responds to devices trying to connect to it, as opposed to "master" mode where it might search for devices to connect to. In essence a device like an Android Phone would be the master which would search for the above "Slave" dongle and connect to it.
- Bluetooth pin code is "1234". This is a "magic password" that will be required for a "master" device like a phone to connect to the dongle. This can be changed by reprogramming the dongle.
- Serial port : Baud = 115Kbps, 8 bits of data, no parity, 1 stop bit
- Serial port flow control disabled.

Programming the device

Using the serial port connections, we can write 'C' code on our Nios computer to send commands to this device to change things such as device name (which *is the name that will appear on your computer/phone/tablet when you start scanning for "discoverable" Bluetooth devices*).

We can also change the "magic password" for any external devices wishing to connect to the "dongle". You will probably have to power down the dongle for any change to take effect when power is next applied. Obviously this means any changes you make are *remembered* by the dongle so you don't have to change them every time you run your NIOS application.

To program the device you will have to send a "$$$" string to it to force it to enter Command mode. The device may time out after a few seconds, returning to data mode if no new commands are sent or if you send the string "---".

Do not send any data for at least 1sec **before** or **after** the "$$$" string otherwise the device will not enter command mode (see section 2.4.1 of the manual), this is to allow $$$ to be sent as "data" if your application required that.

Because there is no default handshaking (CTS/RTS signals) it's probably a good idea to pause a few mS between commands to allow the device to process your commands.

The device may acknowledge each command with a string like "OK" which you can read via the serial port and display using printf() for visual confirmation (or you could just ignore it).

NOTE: Do NOT change the baud rate setting.

The data sheet for the device can be found on connect (checkout "Bluetooth Silver User Manual.pdf"). Chapter 2 of the manual describes sending it commands. This is something you should experiment with, otherwise you won't be able to tell your dongle from every other 391 student's dongle (remember it's wireless – anyone can potentially connect to your device, even from a different room).

Here are some common commands that you can send after the dongle is in command mode. You should NOT experiment with others unless you need to and know what you are doing. Note a "\r\n" is needed after each command including the "---" command but not the $$$ command above.

SF,1
This command restores the device to the factory defaults. Useful if another student has used it before
Example: **SF,1\r\n** // Restore factory defaults

SN,<*string*>
This command sets the device name, where <*string*> is up to 20 alphanumeric characters.
Default: N/A
Example: SN,MyDevice\r\n // Set the device name to "MyDevice"

SP,<*string*>
This command sets the security pin code, where <*string*> is up to 20 alphanumeric characters. Each time the device pairs successfully, it saves the Bluetooth address. The device can store up to eight addresses on a first in first out basis. Using this command also erases all stored pairings. You can use the same value that is already set. You cannot erase the pin code, however, you can overwrite the default pin code.

Default: 1234
Example: SP,0123\r\n // Set pin code to "0123"

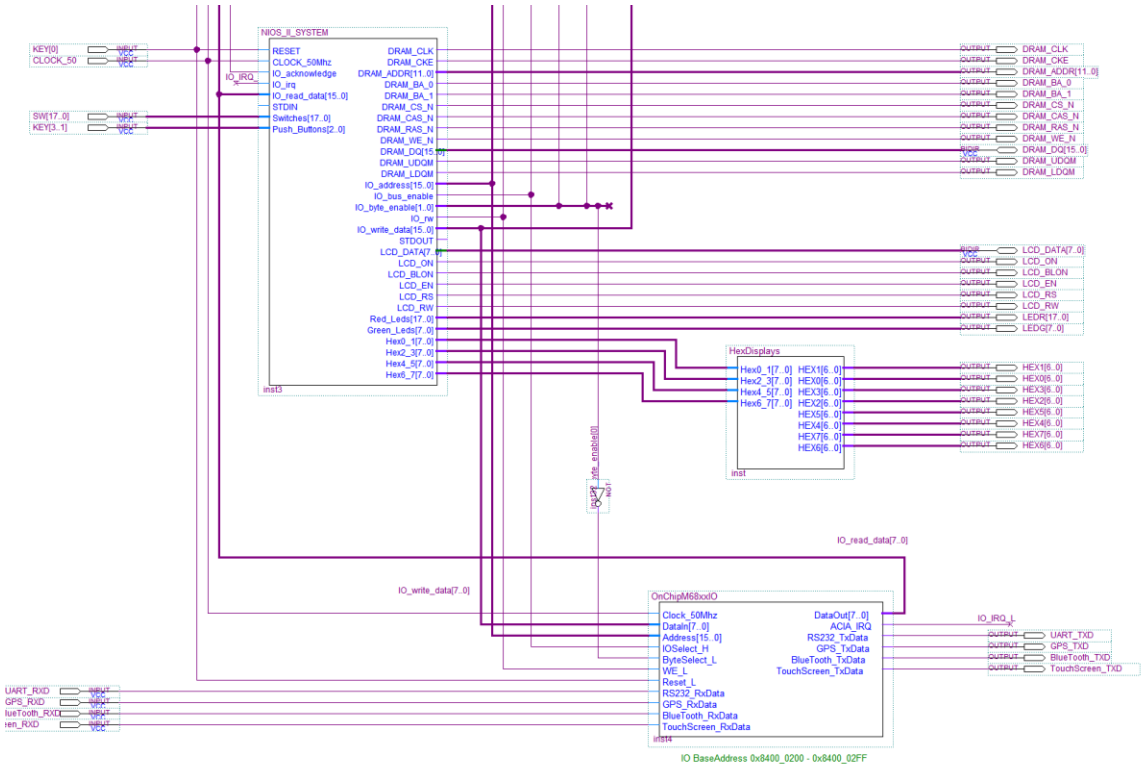
Status Leds

When you apply power to the device, a small **Red** led blinks at 2 Hz frequency. When a Bluetooth master (*such as your phone*) connects to the device then a **Green** Led lights up instead. See if you can use your Android phone to discover your dongle and connect to it, even though no data will be exchanged at this point.

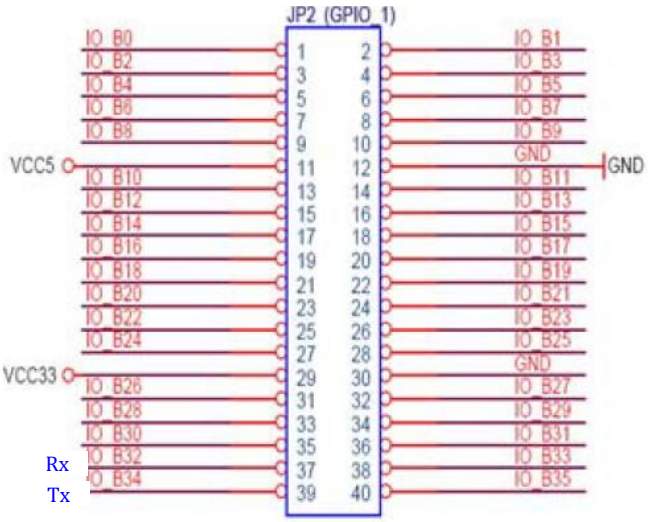
Note for Apple IOS uses: Apple phones do not connect to this device. I understand from discussion on the web that it's an IoS issue. Apple appears to want you to buy special versions of their blue tooth chip sets.

Connecting the Dongle to the DE Board

In exercise 1.3 we created a serial port for the Bluetooth dongle (see below – you should have something similar)



Connecting the Bluetooth Device to the DE Board
At the moment, the Bluetooth_RXD and Bluetooth_TXD pins are hard wired in the Quartus Pin planner to the GPIO_1 connector i.e. the right most (of the two) connector (*you could change these pin assignments if you wish*).



You'll be given special jumper wires to make it easy to connect the DE Board to the Bluetooth chip - make sure you use these. As with the Touch screen and GPS chips, somewhere along the line you'll have to **swap over** the Rx/Tx between DE Board pins above and the actual Bluetooth controller.

Communicating with Bluetooth
Once a master device (such as an Android Phone) connects to the Bluetooth module, communication between DE board UART (i.e. 6850 chip) and the Android phone could not be easier. When the Bluetooth module receives data from Android, it strips off the Bluetooth headers and trailers and passes the data to the DE board UART port. When data is written by the DE board (i.e. NIOS processor) to the 6850 UART port, the Bluetooth module constructs the Bluetooth packet and sends it out over the Bluetooth wireless connection. Thus, the entire process of sending/receiving data wirelessly between the DE/Android host is transparent to NIOS.

Writing the low level Communication routines
As we did in Exercise 1.3, write low level functions to read and write bytes of data to the Bluetooth device. Using these routines, program the dongle to change its name and password.

What do I have to demonstrate?
When you have completed this exercise, show it to the TA for marking.

For this exercise just show your completed Bluetooth dongle wired to the DE board. You should also write a small C program to communicate with the device, get it to enter command mode, change device name and PIN, your program should echo back any characters output by the dongle. Once you have done this, demonstrate that you can connect an Android device to the dongle. That is, that you can discover it (i.e. see the new name for the device) and connect to it with the programmed PIN.